

A

- action handlers 138
 - in jBPM events 145
 - in jBPM exceptions 145
 - in jBPM node type 145
 - in jBPM timers 145
- ActionHandler 146, 166, 186, 201
 - jBPM example implementation 147
- ActionLog 190
- actions. *See* JBoss jBPM, actions
- Active Directory 48
- ActiveMQ 89
 - admin console 93
- actor 139, 162
 - See also* JBoss jBPM, actors
- actorId 162, 176
- adapters 19, 253
- Adobe AIR 174
- Adobe Flex 169
- advantages of open source vs. proprietary software 169
- aggregation 258, 272
- agility 62
- Amazon Mechanical Turk 158
- Amazon.com 9
- anonymous inner class 234
- Ant. *See* Apache Ant
- Apache ActiveMQ 292–293, 298
- Apache Ant 110
 - creating target for running XSD2JavaGenerator 397
 - example of target for generating SDO classes 205
 - using XSD2JavaGenerator example 116
- Apache Axis 4, 10, 270
- Apache Axis 2 10, 286
- Apache BSF 189
- Apache Camel 89, 252
- Apache Commons 257, 299
- Apache Commons Digester 372, 374, 376
- Apache Commons VFS 271, 299
- Apache DBCP 313
- Apache Derby 313
- Apache Felix 53
- Apache Jackrabbit 384
- Apache jUDDI 49
- Apache log4j 119, 192
- Apache Rampart 286
- Apache ServiceMix 41, 252, 256
- Apache Synapse 12, 252
 - advantages over Mule and ServiceMix 44
 - and JMS 271
 - as a proxy service 269
 - as an HTTP proxy 270
 - as message-oriented middleware 271
- case study phase 1 284, 286–298
 - diagram 287
- case study phase 2 285, 299–307
 - diagram 299
- case study phase 3 285
 - diagram 309
- case study phase 4 286
 - diagram 314
- clone mediator 305
 - example configuration 306
- configuration file
 - example 289
- configuring a JMS listener 297
- configuring email transport 303
- configuring JMS transport 293
- creating mock web services 280
- CSV and VFS configuration
 - depiction 307
- custom mediators 273
 - example 301
- database mediator 312–314
 - example configuration 313
- defining JMS queues 292
- description 268
- distinguishing between message and service mediation 274
- enableSec mediator used with WS-Security 317
- enriching data using the database mediator 312
- example of xslt mediator configuration 298
- example using JavaScript 279
- exception handling 303–304
- extendable API 273
- filter mediator 276
- for populating decision services working memory 392

- Apache Synapse (*continued*)
 - high-level architecture
 - depiction 269
 - history 44
 - in and out sequences 269
 - in/out configuration 275
 - incorporating WS-
 - Security 314–317
 - integrating with jBPM 311
 - introducing 268–274
 - introduction to message and
 - service mediation 274–281
 - iterator mediator 311–312
 - example configuration 311
 - example depicted 312
 - JMS mock service 292
 - local registry, example 291
 - local vs. remote registry 291
 - logging 275
 - makefault mediator 290
 - makefault sequence 278
 - managing QoS 314–320
 - marketplace of mediators 301
 - message aggregation 272
 - message mediation
 - example 275–279
 - message transformation 272
 - mock web service 309
 - monitoring and
 - administration 273
 - namespace required for
 - imported file 291
 - POX support 282
 - protocol adapters 270–271
 - QoS and mediation 273
 - registry 286
 - registry element 289
 - regular expressions 276
 - relationship between
 - sequences, services and
 - endpoints 270
 - relationship to WSO2's
 - ESB 44
 - routing and distribution 272
 - service mediation
 - example 279–282
 - service mediator example 289
 - SOAP mock response
 - example 297
 - specifying transport
 - parameters 300
 - strong XML support 272
 - submitting JMS test message
 - in ActiveMQ 298
 - switch mediator 306
 - example 277
 - task mediator requires default
 - message mediator 310
 - example configuration 310
 - when to use it 309
 - tasks and timers 273
 - testing using soapUI 279
 - throttle mediator 317–320
 - and security 318
 - example 319
 - fault block 320
 - uses for 317
 - transport switching using
 - JMS 294
 - using header element 279
 - for VFS configuration 305
 - using named sequences 291
 - using on-fail 304
 - using publishWSDL
 - mediator 316
 - using task and database
 - mediators 308–314
 - using the endpoint
 - element 280
 - using the inSequence and out-
 - Sequence mediators 280
 - using the VFS transport 299–
 - 301
 - using wire-tap message
 - pattern 304–308
 - using WSDL endpoints 294
 - using XML Schema for
 - validation 290
 - using XSLT for CSV file
 - processing 301
 - using xslt mediator to create
 - mock service response 309
 - VFS configuration usage and
 - example 299
 - VFS transport type 300
 - web service mediation 56
 - working with CSV files 301–
 - 303
 - WS-Addressing 316
 - WS-Policy example 315
 - WS-Security, configuration
 - example 316
 - xslt mediator 291
 - example configuration 292
- Apache Tomcat 108, 136
- Apache Tuscany 195
 - @Callback annotation 101
 - @Conversational
 - annotation 97
 - @EagerInit annotation 98
 - @Init annotation 98
 - @OneWay annotation 101
 - @Property annotation 77
 - @Reference annotation 71,
 - 86
 - @Removable annotation 71
 - @Scope annotation 98
 - @Service annotation 71
 - benefits to running as a web
 - application 108
 - binding types supported 87
 - callback implementation 100
 - callback interface 100
 - caveats to using
 - XSD2JavaGenerator 116
 - configuring web.xml 110
 - creating auto-generate
 - WSDL 68
 - creating java interfaces for
 - jBPM action handler 197
 - depiction as interface to
 - jBPM 202
 - deploying to Tomcat 111
 - embedded server 209
 - example component
 - definition 70
 - example of using component
 - type file 96
 - example of using SOAP over
 - JMS 90
 - for service composition 266
 - illustration of distributed
 - architecture 108
 - implementation
 - with BPEL 80
 - with implementation
 - composite 80
 - with OSGi 80
 - with scripting languages 80
 - with Spring 80
 - with XQuery 80
 - integrating scripting code
 - with Java 105
 - integration with Esper 246
 - introducing 53–54
 - Java component
 - implementation 71
 - Jetty built-in web
 - container 108
 - language implementation
 - types 80
 - packaging as a WAR 108
 - properties implementation
 - options 78

- Apache Tuscany (*continued*)
 - relationship between WSDL and SCA Java client interface 197
 - Ruby example 104–108
 - running as embedded container 108
 - SCA and SDO implementation 54
 - SCA graphical depiction using Eclipse 396
 - scripting example using Ruby 105
 - scripting language caveats 104
 - scripting language support 104–108
 - SDOUtil 119
 - sequence diagram illustration for decision service 403
 - service cloud 113
 - simple composite example 67
 - staring SCA embedded domain 83
 - steps for setting up to run within a web application 109–113
 - support for multiple web containers 108
 - used for decision services 329
 - using Ant to create WAR file 110–111
 - using domain manager 113
 - using Eclipse STP plug-in 396
 - using embedded domain 70
 - using JMS binding 89
 - using SDO for complex XML support 114
 - using SOAP binding 89
 - using soapUI for testing 111
 - using XSD2JavaGenerator for decision services 397
 - WAR file for web container deployment 110
 - WAR file packaging 109
 - with ActiveMQ 90
 - XSD2JavaGenerator to generate SDO classes from XML Schema 116
 - Apache VFS 45
 - Apache Xerces 272
 - Apache's ActiveMQ 271
 - API 11
 - Applicant Tracking System 210, 262
 - Aqualogic 54
 - ArrayList 347
 - assignment 162, 165
 - AssignmentHandler 146, 163, 165
 - asynchronous
 - communications 99
 - See also* JBoss jBPM asynchronous continuations
 - asynchronous messaging 45
 - benefits 264
 - depiction 265
 - ATOM 391
 - Atom/RSS 51
 - audit logging 180
 - audit vs. debug type logging 192
 - auditing 256, 391
 - automation 157
 - autonomous services 391
 - availability 261
 - avg 237
 - Axis2 89
 - axis2.xml 303
- B**
-
- BAM 21, 99, 147, 153, 182, 190, 220, 274
 - and ESP, differences 220–221
 - role vis-a-vis ESP 220
 - BeanShell 138, 142, 144, 166, 168, 181
 - example usage in jBPM 189
 - expression 189
 - scripting in jBPM 187–189
 - vs. standard Java 189
 - Bernhardt, Thomas 47, 221
 - best practices 266
 - BI 46
 - binary optimization 272
 - BinarySecurityToken 271
 - binding.ws 199
 - BIRT 47
 - Blackberry 221
 - Blue Titan 56
 - BPDM 136
 - BPDL. *See* WS-BPEL
 - BPM 14, 26, 237
 - and CEP 134
 - and ESP 220
 - and event driven architecture 134
 - and events 128
 - and wait states 129
 - as a paradigm shift 134
 - as executable processes 126
 - as secret sauce 17
 - benefits 17
 - best practices vis-a-vis ESB 129
 - differences to ESB 129
 - relationship to ESB 129
 - compared to ESB 19
 - contrast to conventional applications 30
 - definition 16
 - development lifecycle 130
 - evaluation 30
 - goal and objectives 126
 - hydration and dehydration 19, 134
 - ingredients 31
 - initiating a process instance 133
 - open source products 32
 - relationship to services and components 128
 - relationship to SOA 16–17, 127
 - selecting a product 34
 - slow adoption 134
 - still requires developer expertise 132
 - used to accelerate new products and services 126
 - using business rule engine 330
 - why not to include technical details in process definition 145
 - BPM, evaluation 35
 - BRE 17, 27
 - BRMS 18
 - authoring IDE 36
 - benefits 332
 - definition 332
 - introduction 332
 - rule engine relationship depicted 332
 - testing using soapUI 403
 - See also* EDM
 - Brynjolfsson, Erik 126
 - bus topology 254
 - business rules, ownership 327
 - business agility. *See* BPM
 - business analyst 327
 - business events. *See* events
 - Business Intelligence. *See* BI
 - Business Process Management. *See* BPM

business rule
 declarative in nature 326
 definition 326
 Business Rule Approach 327
 Business Rule Engine. *See* BRE
 Business Rule Management System. *See* BRMS
 business rule revolution 35
 business rule service. *See* decision service
 business rules 157
 archiving 385
 as assets 328
 as enterprise asset 18
 as when/then constructs 330
 benefits 328–329
 business alignment with IT 329
 dangers in hard-coding in applications 390
 definition 18
 examples 326
 for compliance 329
 how to harvest 327
 pattern matching 330
 relationship to composite applications 390
 relationship to SOA 329
 revolution 36
 rule versioning 388
 understanding 326–328
See also BRMS
See also decision services
 Business Rules Management System. *See* BRMS

C

C# 10
 caching 262
 callbacks 316, 341
 with event stream processing 100
 cancel-timer 165, 168, 188
 case 237–238
 cast 237–238
 Castor 113
 centralized router 259
 CEP. *See* ESP
 Chodavarapu, Chodavarapu 314
 Choicepoint 219
 choreography vs orchestration 41
 Cisco 55
 clone 272

clone mediator 285, 305, 320
 closure 105
 clustering 11
 coalesce 237
 code generation 11, 397
 command executor. *See* JBoss jBPM, command executor
 Common Object Request Broker Architecture. *See* CORBA
 Complex Event Processing. *See* ESP
 compliance 167, 242, 328, 391
 component
 definition 62
 distinguishing from a service 62
 how it becomes a service 63
 component framework. *See* OSGi; *See also* SCA
 component-based routing 260
 components 52
 and composites 23–24
 composite
 components 64
 recursive 66
 SCA definition 66
 composite applications, role of business rules 330, 390
 composite services 13, 24, 183, 253
 composition 63
 compound value
 restrictions 349
 compression 271
 conditional expressions 334
 conditions 330
 Configuration.configure 227
 connection pooling 313
 connectors 253
 consequences 330
 content filter 262
 content filtering 262
 content switch 55
 content-based routing 260
 conversational services 94
 converters, in jBPM 155
 Cooper, Peter 105
 CORBA 5, 7, 40
 IDL 8
 correlation 245
 rules 272
 count 237
 CreateOrderService 293
 createProcessInstance 202, 210
 create-timer 165, 188

CRM 254
 cron 261, 273
 cross-cutting concerns 274
 CSV 253, 261, 265, 285, 299
 processing using Apache Synapse 302
 custom mediators 273, 285
 CVS 385
 CXF 10

D

dashboards 21, 46
 data flow 267
 data validation 114
 database mediator 285
 database query 314
 databases 130
 DB mediator. *See* database mediator
 db4objects 212
 dblookup 313–314
See also Apache Synapse database mediator
 dbrecord 313
See also Apache Synapse database mediator
 DCOM 5, 7
 decision node. *See* JBoss jBPM, decision node
 decision service 327
 depiction of populating working memory 393
 designing 392–396
 high-level depiction 391
 decision services
 and BRMS 391
 benefits and value 390
 characteristics 390
 decision operation 394
 implementation using SCA 400–403
 sequence diagram 403
 designing the WSDL 393–396
 facilitates reusability 329
 load operation 394
 implementation using Java 399
 implementation using SCA 397–400
 loading working memory 397
 relationship to SOA 365
 requires SOA 329
 resume operation 394

- decision services (*continued*)
 - supporting multiple rule domains 394
 - suspend and resume operations implementation using SCA 403
 - suspend operation 394
 - using JBoss Cache to populate working memory 392
 - using SCA/Apache Tuscany 396
 - with SOAP over HTTP 396
 - WSDL depiction 396
- decision tables 362
- DecisionHandler 142, 146
- decisions and business rules 325
- decomposing 235
- dehydration 134
- delivery assurances 271
- dependency injection 76, 147
- deployment snapshots 384
- digital signatures 314–315
- distributed architecture 108
- distributed computing 4–6
 - contrast to SOA 9
 - IBM System/360 4
 - socket programming 4
- Distributed Computing Object Model. *See* DCOM
- distributed ESB
 - environment 267
- dot notation 228
- Drools 333, 338–339
 - activating the rule engine 375
 - activation-group attribute 342, 370
 - agenda-group attribute 343
 - agenda-group attribute example 343
 - alternatives to DRL 358–363
 - applying logical conditionals to fact objects 350–351
 - auto-focus attribute 343
 - example 343
 - benefits of business rules 328–329
 - case study overview 365–366
 - cautions for using Java methods 355
 - caveats to using functions 342
 - comparing RuleFlow to JBoss jBPM 359
 - compound value restrictions 349
 - conditions and pattern matching 346, 369
 - contains operator 348
 - creating a RuleBase 372
 - creating new working memory example 373
 - date-effective attribute 344
 - example 344
 - date-expires attribute 344
 - example 344
 - decision table
 - example depiction 362
 - decision table configuration 362
 - dialect attribute 344
 - DRL 335
 - constructs depicted 339
 - example 368, 370
 - header elements 340–342
 - overview 339–355
 - DSL
 - compared with DRL rule 378
 - editing using Eclipse plug-in 361
 - example 359–361
 - for business users 377
 - getting started 359
 - rule example 360
 - templates 359, 379
 - duration attribute 344
 - Eclipse plug-in 334, 371
 - caveats for using 386
 - examples
 - a function 342
 - accumulate
 - conditional 353, 369
 - and conditional 350
 - collect conditional 352
 - eval conditional 351
 - exists conditional 351
 - forall conditional 352
 - from conditional 352
 - insertLogical method 354
 - not conditional 351
 - or conditional 351
 - running embedded 338, 371–377
 - expander keyword 341
 - fact aliases 336
 - fact object example 334
 - facts and working memory 328
 - feature overview 334
 - global keyword 341
 - global variables 341
 - Guvnor BRMS 334
 - history 39, 333
 - import keyword 335, 340
 - in and not in group conditionals 349
 - insert and insertLogical methods 354
 - instantiating RuleFlows 375
 - integrating with Esper 341
 - integrating with Java using functions 341
 - iterator and collection-related conditionals 352–353
 - listeners for debugging and troubleshooting 374
 - lock-on-active attribute 345
 - example 345
 - major releases 39
 - managing flow control using agenda groups 343
 - managing rule activations using the lock-on-active attribute 345
 - managing rule priority using salience 346
 - matches operator 348
 - memberOf operator 348
 - no-loop attribute 346
 - operators using multiple restrictions shorthand 347
 - package keyword 335, 340
 - packages used as namespaces 381
 - pattern operators 347–353
 - patterns and constraints 346–347
 - preventing recursive looping 346
 - querying facts in working memory 356, 371, 376
 - retract keyword 337
 - retract methods 355, 377
 - role of the consequence part of rules 354
 - Rule Agent 384–385
 - rule attributes 342–346
 - rule consequence/then 341
 - rule construction 335
 - ruleflow-group attribute 346, 357, 368, 370
 - RuleFlows 368
 - depiction using Eclipse plug-in 356, 375
 - diagram for case study 367
 - example 358

Drools, RuleFlows (*continued*)
 reasons for using 356–357
 running embedded instead of using decision service 371
 salience attribute 337, 343, 346, 368–370
 soundlike operator 349
 steps involved when running embedded 371
 support for Java fact object introspection 347
 system properties 347
 understanding activation order 336
 understanding the consequence/then rule part 354–355
 understanding when conditions 346–353
 update example 354
 update method 354, 369, 371
 use of MVEL 347
 using decision tables 362–363
 using DSL for natural language rules 358, 378
 using functions example 342
 using Rule Agent for loading rules 398
 using RuleFlow 356–358
 using rules for computations 371
 variable binding 353
 example 368
 web service enabling 39

Drools Guvnor
 administration model view 385–386
 archiving rules 385
 assets view 380–381
 assigning metadata to rules 388
 binary packages and the Rule Agent 384
 business analysts ideal users 386
 business rule assets 382
 creating a package 381
 creating binary deployment snapshot 384
 creating new rule assets 381
 decision table uploading 387
 decision table web editor 387
 DRL rule editor depiction 387

DSL configuration assets 383
 DSL text editor 387, 389–390
 enumerations 383
 fact model required for rule authoring 381–382
 five rule authoring options 386–387
 function assets 383
 functionality overview 379–386
 guided editor 386 depiction 388
 importing and exporting 385
 managing permissions 386
 managing release process 385
 managing rule categories 385
 managing status categories 385
 model assets 383
 other assets 383
 package naming conventions 382
 packages module view 381–383
 rule authoring 386–390
 rule categorization 380
 rule status 381
 rule versioning 388
 RuleFlow assets 383
 snapshot as point-in-time configuration 385
 technical rule assets 382
 test scenario assets 383
 testing support 403
 types of rule assets 382
 typical users 380
 user permissions 380
 using enumerations in authoring 388
 validating a package's configuration 383
 validating rules 387
 viewing rule source in authoring environment 389
 Web 2.0 interface 379
 XML property assets 383
See also decision services

Drools Rule Language. *See* Drools, DRL
 drools.setFocus() 343
 DroolsDecisionService 396
 DSL 327
 DTO 366
 duedate 166

E

EAI 254
 ESB architectural alternative 255
 MOM architecture example 254–255

eBay 9

ebXML 63, 284

Eckerson, Wayne 21

Eclipse 34, 52, 65, 204
 jBPM plug-in, getting started 146–147
 plug-in 334

EDA 45, 220
 and BPM 134
 definition 221
 relationship to SOA 221

EDI 261, 284
See also electronic interchange

EDM 26
 components 36
 defined 333, 390
 illustration 18
 open source evaluation 35–39
 open source products 37
 relationship to BRMS 333
 relationship to SOA 17–19
 role in Open SOA Platform 35
 role in SOA 36

EJB 5, 7, 391

electronic interchange 253

Email.rb 105

enableSec 317

encryption 262, 315

endpoint 259, 269, 280

endpoints 391

enrichment 258

Enterprise Application Integration. *See* EAI

Enterprise Decision Management. *See* EDM

Enterprise Java Beans. *See* EJB

Enterprise Resource Planning. *See* ERP

Enterprise Service Bus. *See* ESB

EPAdministrator 231, 241

EPL. *See* Esper, EPL basics

EPRunTime 231

EPServiceProvider 222, 229, 231, 246, 341

EPStatement 226

EQL 21, 47

Equinox 53

- Erl, Thomas 24
- ERP 5, 254, 365, 392
- ERRunTime 231
- ESB 12
 - adapters 257
 - and long transactions 267
 - API support for
 - extensibility 263
 - appropriate uses 263–265
 - as a SOA enabler 253
 - as message-oriented middleware 257
 - based on bus topology 255
 - based on JMS messaging backbone 255
 - BPM's role contrasted 267
 - chain or pipeline for
 - transformations 261
 - compared to BPM 19
 - connectivity is key 256
 - core capabilities 256–263
 - cost compared to EAI 255
 - data or microflow 19
 - depiction of central role 39
 - depiction of policy management 262
 - distributed environment 267
 - distribution contrasted with routing 260
 - example of configuring an adapter 257
 - features and capabilities 19
 - for asynchronous messaging 264
 - for event processing
 - depiction 258
 - for metering 262
 - for quality of service (QoS) 261
 - for routing and distribution 259–260
 - high-throughput processing 268
 - history 253–256
 - illustration where performance problems may exist 268
 - in the extranet 253
 - inappropriate uses 265, 267–268
 - message transformation 260
 - metrics to monitor 263
 - monitoring and administration 262
 - open source evaluation 39–45
 - open source products 42
 - patterns to routing 259
 - performance bottlenecks
 - introduced by using 268
 - proper role and use 129
 - protocol adapters 256
 - protocol bridging 39, 265
 - relationship to SOA 19–20
 - role in Open SOA Platform 39
 - routing and transformation 129
 - secure network configuration
 - depiction 266
 - service enablement 264
 - service virtualization 264
 - splitter pattern 311
 - supporting legacy protocols 308
 - tasks and timers 261
 - typical architecture
 - depicted 255
 - typical role of JMS 257
 - used for service mediation 260
 - using web mediation to improve QoS 262
 - when to use BPM 260
 - wire-tap pattern
 - described 304
 - XML as lingua franca 255
 - XML-based messaging 258
 - See also* Apache Synapse
 - esbsite.org 301
- ESP 147, 190, 258, 391, 404
 - and BPM 128
 - as delivery channel for BAM 220
 - definition 217
 - diagram depiction of ESP at work 219
 - for compliance 219
 - for deriving instant insights 218
 - for detecting patterns 218
 - for fraud detection 219
 - function of the processor 219
 - open source evaluation 45–47
 - real-time monitoring for performance and compliance 218
 - relationship to BAM 21, 46
 - relationship to messaging 46
 - relationship to SOA 21–22
 - role in Open SOA Platform 45
 - used to detect opportunities or threats 218
 - See also* Esper
- Esper 190, 304, 306, 404
 - accessing variable using Configuration class 232
 - advanced features 237
 - advantages to using subscribers over listeners 226
 - and JavaBeans 224
 - Apache Synapse case study 285
 - as an ESP engine 224
 - case study 223
 - caveat when working with imports 241
 - chaining view types 234
 - configuring JDBC datasource 244
 - using XML configuration 244
 - configuration options 226–227
 - correlation 237
 - create window clause 235
 - creating named windows 235
 - defining and registering queries 225
 - differences between normal event streams and named windows 236
 - embedding SQL in EQL 244
 - EPL basics 227–237
 - comments 228
 - filter ranges 228
 - from clause filtering 228
 - functions 237–241
 - query example 227
 - querying 227–231
 - where clause 228
 - event object aliases 226
 - every operator 242
 - examples
 - combining the max, prev and sort functions 238
 - creating new event stream 235
 - publishing events to 230
 - querying against a named window 236
 - registration and subscriber class 229
 - user-defined function 240
 - using getter methods to cast event properties 225

- Esper, examples (*continued*)
 - using sort function 238
 - using SQL in EQL 245
 - exposing as SOAP service 246
 - getting started 224–227
 - how patterns differ than normal select statements 241
 - illustration of service enabling with SCA 246
 - insert into clause 235
 - inserting new streams 235–236
 - installation 224
 - instanceOf function 238–239
 - to determine object type 238
 - used with case and cast 239
 - introducing 47
 - Java methods for use as user-defined functions 239
 - JDBC 243
 - listeners and subscribers 226
 - managing session 247
 - named window retention criteria 236
 - named windows 234
 - output clause 234
 - outputting JDBC data into event stream 245
 - pattern matching 225
 - patterns 241–243
 - correlation 242
 - defining 241
 - prev function 237–238
 - reasons for dynamically creating new streams 235
 - receiving events from Apache Synapse 304
 - receiving events from rule engine 341
 - referencing JDBC columns in EQL 245
 - registering EPL statements 247
 - SCA components 246
 - SCA composite file configuration 249
 - service enabling 245, 249–250
 - with SCA 223
 - snapshot clause 234
 - sorted window 237
 - steps involved in preparing 224
 - testing services using soapUI 250
 - time window 233
 - example 233
 - timer
 - at pattern guard 243
 - interval pattern guard 243
 - within pattern guard 242
 - understanding event objects 224–225
 - understanding pattern guards 242–243
 - use with decision services 391
 - user-defined functions 239–241, 244
 - using auto-import to import Java classes 240
 - using filters with named windows 236
 - using JDBC to pull SQL data 244–245
 - using junit for testing 232
 - using the pull API 226
 - using to filter jBPM events 234
 - variables 231–233
 - how they can be created 231
 - reasons for using 231
 - updating 232
 - usage in EPL statements 232
 - used for static value assignment 233
 - views 233–234
 - win,length function 233
 - working with properties 224
- Esper EPL
- similar to SQL 225
 - used to define patterns 225
- Esper Processing Language. *See* Esper, EPL basics
- EsperIO 245
- EsperManager 248
- EsperManagerImpl 248
- EsperManagerService 249
- Espertech 47
- ETL 261
- eval 342
- evaluation priority 346
- event 219
 - alerting 134
 - buckets 236
 - correlation 233
 - defining 46
 - identifying expected but non-occurring event 243
 - monitoring and BPM 134
 - object 219
- event architecture 21
- Event Driven Architecture. *See* EDA
- event processor. *See* ESP
- Event Query Language. *See* EQL
- event stream 219
- Event Stream Processing. *See* ESP
- EventBean 226
- EventNotificationComponent 100
- events 218
 - and BPM 128
 - and pub/sub messaging models 258
 - definition 21
 - ESB as propogator 285
 - in JBoss jBPM 151
 - in the enterprise 218–219
 - jBPM superstate 182
 - monitoring 220
 - pattern matching 241
 - See also* ESP
- exception handling 144, 180
 - in jBPM 185–187
- ExceptionHandler 146
- exception-handler 138, 188
- execute method. *See* ActionHandler
- ExecutionContext 146
- expressions. *See also* jBoss jBPM, action expressions
- extensibility 258
- Extract, Transform and Load *See* ETL
- extranet 253, 266

F

- F5 Networks 55
- fact constraints 346
- fact objects 334
 - qualifiers 336
- FactHandle 377
- facts, defined 328
- fact-sets 327
- failover 262
 - processing 273
- fault management 272
- file-drop 285
- filter mediator 276
- Finger, Peter 17, 126
- Fiorano 252
- fireAllRules 376

firewall 273, 318
 Forgy, Charles 330
 fork. *See* JBoss jBPM, fork
 FOSS. *See* open source
 FQN 392, 399
 fraud-detection 242
 FTP 5, 19, 253, 265–266, 284–285
 fully qualified names. *See* FQN

G

Galaxy. *See* MuleSource Galaxy
 Ganymede 65
 gateway 12
 getObject 202
 global variable 341
 Globally Unique Identifier. *See* GUID
 Goldberg, Larry 35
 Google GWT 169, 379
 governance 16, 26, 48, 55
 Graham, Ian 326, 330
 Graph-Oriented Programming 136
 GraphSession 171
 findAllProcessDefinitions() 171
 Groovy 104, 263, 272
 GUID 301
 Guvnor. *See* Drools Guvnor

H

handlers, assignment 163
 See also action handlers
 hardware routers 273
 Harvard Business Review 126
 hello world 275
 HelperContext 119
 Hibernate 35, 153, 171, 174, 208
 hibernate.cfg.properties 170
 hibernate.cfg.xml 170
 hibernate.properties 170, 204
 HireRight 286
 Hohpe, Gregor 259
 HTTP 40, 195, 253, 271, 284
 headers 11
 proxy 279
 hub-and-spoke 254
 human intelligence tasks 157
 hydration 134

I

IBM 9, 42, 53, 254
 IBM WebSphere XE 334
 IMAP 271
 inference engine. *See* BRE
 InformationWeek 9
 Infravio 44
 in-memory database 392
 inSequence 280, 320
 instanceof 237–238
 integration brokers. *See* EAI
 Intel XML Content Router 55
 interoperability 258
 intranet 253
 introspection 347
 inversion of control 53, 84, 93
 IONA 53, 56
 IP address 286
 IP filtering 317
 iPhone 221
 isAllow 319
 issueMgmt-distributed.composite 112
 iterator mediator 286
 itinerary 267

J

JAAS 386
 JasperReports 47
 Java 10
 annotations 95
 helper classes 63
 scripting. *See* BeanShell
 Java Authentication and Authorization Service *See* JAAS
 Java Business Integration. *See* JBI
 Java Servlet API 121
 JavaBean 78, 86, 227
 JavaScript 104, 263
 Java-WS 10
 JAXB 114
 Jaxen 275
 JBI 41, 93
 JBoss 9, 392
 jPDL 267
 jPDL compared with BPEL 137
 See also RedHat
 JBoss Cache 392
 configuration 398
 configuring with Apache Tuscany 396
 initializing the cache 399
 use in decision services 392
 JBoss Drools. *See* Drools
 JBoss Enterprise Middleware (JEMS) 334
 JBoss ESB 252
 JBoss jBPM
 acquiring a logging instance 191
 action handler touchpoints 145
 ActionHandler sample implementation 147
 actions 145–151
 as integration points 145
 actor assignment options 164
 actors 139, 162
 in mail node 140
 pooled 162
 and CEP 134
 API 136
 as the glue within a business process 144
 assignment, dynamic 163
 asynchronous continuations 136, 192–194, 213
 depicted 193–194
 how to add to a node 193
 required jBPM Enterprise 193
 where they can be used 194
 audit logging 190–192
 BeanShell
 context environment 188
 example 142
 expressions in decision node types 189
 expressions vs. scripts 189
 scripting 187–189
 benefits to using as a service 196
 command executor 193
 configuring a database 170
 Console 132, 135, 143, 162, 183, 211
 and identity 163
 screenshot 136
 task form limitations 169
 timers 166
 contrasted to Drools RuleFlow 359
 creating a process object for Esper 223
 cycling through available logs 191

- JBoss jBPM (*continued*)
 - decision node
 - options 142
 - type 142
 - type expression
 - example 187
 - default thread blocking
 - behavior 193
 - depiction of logging events to Esper via SCA 223
 - deploying business
 - processes 132
 - deployment options 132
 - developing the listProcesses
 - service operation 203–210
 - development lifecycle 130–136
 - differences between state and node node type 139
 - differences between task-node and task 160
 - disabling logging 190
 - Eclipse Deployment View 132
 - Eclipse editor example 131
 - Eclipse plug-in 35
 - email support 136
 - embedding 132, 136
 - API in clients creates tighter coupling 195
 - Enterprise edition 182, 193–194
 - event objects 151
 - events 138, 151–153
 - example of retrieving logs for a given process
 - instance 191
 - examples of business
 - processes 131
 - exception handlers defined at root of process
 - definitions 187
 - exception handling 138, 185–187
 - don't use for controlling flow 185
 - limited to custom handlers 185
 - exposing API as protocol neutral services 195
 - exposing as services lowers learning curve 195
 - fork and join nodes 142–144
 - Graphical Process Designer (GPD) 131, 160, 170, 181
 - history 34–35, 129
 - identifying good candidate
 - processes 131
 - illustration of how it can be used with external
 - services 196
 - initiating a process
 - instance 133
 - integrating with Apache Synapse 311
 - integrating with Esper with custom logger 222
 - Java API 160, 167, 169, 184, 195, 208
 - identifying processes 170–172
 - using Hibernate to query database 172
 - using to complete a task 177–178
 - using to find open tasks 174–175
 - using to find pooled tasks assigned to user 176
 - using to find tasks assigned to a user 176
 - using to retrieve process instances 172–174
 - using to retrieve processes 170–172
 - jbpm.mail.templates.xml file
 - for customized messages 141
 - Job Executor for asynchronous continuations 193
 - jPDL example 137, 146, 148
 - of action handler
 - definition 200
 - of exception handling 185
 - of superstates 181
 - of timers using reminders 167
 - swimlane 164
 - using events 152
 - using expression
 - attribute 150
 - jPDL not a standard 136
 - jPDL subprocess example 184
 - listProcesses sample SOAP request 204
 - local variables 154
 - when to use 155
 - logfile types 190
 - mail node 140–142
 - attributes and elements 140
 - mail template variable
 - properties 141
 - monitoring with Esper 220
 - node nodetype 137–139, 193
 - nodes 137–144
 - action 138
 - and states 132
 - purposes 137
 - transitions 138–139
 - type attributes and elements 138
 - types 137
 - operations exposed as
 - services 202
 - Process Archive file (PAR) 132
 - Process Definition Language (PDL) 136
 - process instance status 135
 - process instance variables, retrieving from action handler 201
 - process variable example 146
 - process variables 153, 168
 - as executable Java objects 149
 - propagated to subprocesses 184
 - properties 147
 - using XML data 148
 - property configuration
 - types 148
 - property instantiation 148
 - reasons for service-enabling
 - using SCA 201
 - RuleFlow implementation in Drools 357
 - SCA client implementation
 - class 198
 - SCA composite file
 - example 198
 - searching by instance key 135
 - service enabling 181
 - specifying runtime
 - components 132
 - state node 139–140
 - state node type 183
 - state vs. node for asynchronous wait states 193
 - steps in setting up SCA client 196–201
 - subprocess diagram
 - example 183
 - best practices 184
 - runtime binding 184

- JBoss jBPM (*continued*)
- subprocesses 180, 183–185, 212
 - and reusability 183
 - for decomposition 183
 - superstates 181–182, 212
 - as a grouping of nodes 181
 - supporting different process definitions as a service 212
 - swimlanes 164–165
 - process diagram
 - depiction 164
 - used for roles 164
 - tasks 135
 - API 158
 - assignment 161
 - basics 158–161
 - controllers 168
 - element and
 - attributes 160–161
 - element
 - configuration 160–161
 - expressed in jPDL 158, 162
 - management using jBPM
 - Console 159–160
 - nodes 139
 - user management 161–165
 - using the API 169–178
 - variables 168
 - timers 135, 138, 165–168, 182
 - as critical functionality 167
 - elements and attributes 166
 - only available in
 - Enterprise 165
 - token signal 140
 - tokens 139
 - in joins 143
 - transitions 144
 - elements and attributes 144
 - path options 144
 - understanding release
 - editions 136
 - use LoggingSession instead of
 - LogginInstance 191
 - uses of state node 140
 - using a mail template
 - example 141
 - using action expressions 149–151
 - using actions for inserting
 - programming logic 145
 - using actions to abstract
 - complexity 145
 - using actions with events 151
 - using converters 155
 - using db4objects to store process data 212
 - using JMS to instantiate business process 286
 - using SCA for client API 196–201
 - using variables 153–155
 - variable types 153
 - VariableInstance 155
 - variables 225
 - variables and tokens 155
 - wait states 137, 139
 - where BeanShell script can be used 188
 - why use web services for callouts 195
 - why using it with SCA is so compelling 196
 - JBoss jBPM Suite. *See* JBoss, jBPM
 - JBoss Rules. *See* Drools
 - JBossWS 10
 - jBPM
 - as graph oriented programming 136
 - Console 130
 - using assignment handlers 163
 - jbpm.cfg.xml 141, 190
 - jbpm.composite 208
 - jbpm.mail.templates.xml 141
 - jbpm.wsdl 204, 211
 - JBPMClientMain 250
 - JbpmContext 170, 173, 176, 178, 191
 - JBPMHelper 170, 177
 - jConsole 263
 - JDBC 227, 237
 - using with Esper to pull events 244–245
 - Jess 330
 - Jetty 108
 - JiBX 113
 - Jitterbit 26
 - licensing restrictions 43
 - JMS 12, 22, 40, 74, 266
 - for enterprise jBPM 193, 210
 - mock service 292
 - relationship to SOA 22
 - role in an ESB 257
 - support in Apache
 - Synapse 252, 264, 271
 - used in Apache Synapse case study 286
 - using with Drools 391
 - using with Esper 245
 - when to use pub/sub model 257
 - JMX 247, 249, 262, 273
 - JRuby 104
 - jRuby 263
 - JSON 87, 391
 - JUnit 132, 403
 - creating test cases for
 - Esper 228
 - test case example for
 - Esper 230
 - jUnit 228
 - Jython 263
-
- K**
-
- Kanneganti, Ramarao 314
 - Knopflerfish 53
 - knowledge retention using business rules 328
 - Krishnan, M.S. 17, 126, 218
-
- L**
-
- LDAP 15, 22, 47, 164
 - Leaps 330
 - left-hand side 354
 - listActorTasks 202
 - listeners 226
 - listInstanceTasks 202
 - listInstanceTokens 202
 - listProcesses 203
 - creating SCA composite file 208
 - creating SCA Java implementation 207
 - creating SCA Java interface 206
 - generating SDO binding classes 205
 - steps for creating 204
 - WSDL and XML Schema depiction 205
 - listProcessInstances 203
 - listservices.composite 209
 - load balancing 11, 262, 273
 - localEntry 291
 - log4j.properties 120
 - logging 256, 269
 - LoggingInstance 190, 192
 - caveats in using 191
 - LoggingSession 190, 192
 - logical operators 348

M

mail transport 285
mailto 304
maintenance costs reduced
 using a rule engine 328
makefault 278
makefault mediator 290
management 256
mapUBLOrder.xslt 291
marshaling 208
max 237
MaximumConcurrentAccess
 319–320
McAfee, Andrew 126
MD5 317
message
 channels 254, 265
 injector 273
 mediation 270
 splitting 285
 throttling 273
 transformation 39
MessageID 301
MessageInjector 286, 310
Message-Oriented Middleware.
 See MOM
messaging system. *See* ESB
metering 262, 273, 314
micro-flow 267
Microsoft 9, 28
Microsoft Excel 334
middleware 264
MIME-type 301–302
min 237
mock services 280
MOM 254
MQ Series 40
MTOM 272
Mule. *See* MuleSource Mule
MuleSource 51
MuleSource Galaxy 16, 48
MuleSource Mule 41, 252, 263
MVEL expression language 347
MySQL 227

N

named query 371
NEsper 222
nested accessor 347
.NET 222, 246
 integrating with jBPM using
 SOAP 201
New Age of Innovation, The 126

NFS 5
node 137
 See also jBPM node
NodeLog 190
nodetypes. *See* JBoss jBPM
Novell's Nsure 49

O

OASIS 54, 63
OASIS UBL 261, 284
object database 212
Open Office 334, 363
Open Service Oriented
 Architecture 64
Open SOA Platform 218
 components 16–25
open source
 community 28
 evaluation criteria 30
 hidden documentation 56
 Open SOA Platform 29
OpenDocument 63
OpenSpan 26
operational decisions 390
 See also business rules
operational monitoring 218,
 258
Oracle 9, 53, 227
orchestration 158
 vs choreography 41
org.jbpm.context.log 190
org.jbpm.context.log.variableins
 tance.* 190
org.jbpm.graph.log 190
OSGi 52
OSGi Alliance 53
OSOA. *See* Open Source Ori-
 ented Architecture
outSequence 280, 320

P

Package 338, 373
PackageBuilder 338, 361, 373
packages in Drools 382
Pareto 202
password hash 316
PasswordDigest 271, 314
pattern matching
 algorithms 330
pattern restriction 369
patterns 225
 See also Esper patterns

Pentaho 47
Perl 104
personalization using business
 rules 329
Pion 47
pipeline 259
Pointillism 63
PojoCacheFactory 399
policy management 273
pooled actors 162, 164
POP3 40, 271
POX 195, 253, 258
Pralhad, C.K. 17, 126, 218
prev 237
pricing engine 365
ProblemManagementComposite
 67–68
 adding conversational
 support 98
 createTicket SOAP
 operation 75
 CreateTicketComponent 67
 adding conversational
 support 98
 CreateTicketComponentImpl
 Java implementation 72
 creating custom WSDL 87, 91
 Email.rb Ruby example 105
 example incorporating Ruby
 script 106
 IssueManagementComposite
 definition 82
 explanation 80
 using implementation
 composite 82
 Java implementation
 SDOClientImpl 117
 Java interface
 ProblemTicketComponent
 SDO 116
 Java JMSClient interface 92
 ProblemTicketComponent 67
 exposed as web service 68
 Java implementation 71
 Java interface 71
 using properties 76
 ProblemTicketComponent2
 alternative
 configuration 75
 ProblemTicketComponent-
 Impl using properties
 implementation 77
 ProblemTicketComponent-
 Impl2 alternative
 configuration 75

ProblemManagement-
 Composite (*continued*)
 reference usage example 86
 running as a web
 application 108
 running as web
 application 112
 sequence diagram 72
 SOAP WSDL 73, 75
 SystemErrorComponent 80
 SystemErrorComponentImpl
 Java implementation 83
 using component types 95
 using SDO 115
 using WSDL for port and service
 definition 89
 problemMgmtSDO.composite
 117
 ProblemService.wsdl 115
 ProblemServiceComponent,
 using scripting
 language 107
 ProblemTicketService
 packaged as WAR file 111
 with callback support 104
 Process Archive file. *See* JBoss
 jBPM
 process execution engine 130
 ProcessDefinition 171, 208
 ProcessEvent 223–224, 228
 processId 173–174
 processing pipeline 259, 265
 ProcessInstance 172, 177, 191
 Proctor, Mark 358
 Progress 41, 252
 ProhibitedTimePeriod 319
 properties
 instantiating jBPM 148
See also SCA properties
 protocol adapters, using
 Synapse 270–271
 protocol mediation 55
 protocol switching 12, 285, 288
 protocol transparency 86
 protocols 253
 proxy 12, 25, 56
 pub/sub 22, 257
 publishWSDL 316
 PWS Callback 316
 Python 104, 263

Q

QoS. *See* quality of service
 quality of service 16, 261–262

using Apache Synapse 314–
 320
 Query 173–174
 QueryResults 377
 queues 257, 264

R

Raden, Neil 329
 real-time systems 268
 RedHat 39, 50
 reflection 224, 227, 353
 registry
 metadata 23
 open source evaluation 47–52
 open source products 49
 public or private 15
 relationship to SOA 22
 role in Open SOA
 Platform 47
See also WS02 Registry
 Remote Method Invocation. *See*
 RMI
 Remote Procedure Calls. *See*
 RPC
 repository. *See* registry
 Resin 136
 REST 10, 25, 36, 55, 195, 258,
 391
 combined with WADL 10
 vs SOAP 10
 Rete 328, 330
 retract 337
 reusability 13
 right-hand side 354
 RightNow Technologies 286
 risk mitigation 218
 RMI 24
 RosettaNet 284
 routing 255, 258, 269
 routing rules 272
 routing slip 259
 Roy Shulte 7
 RPC 5–6, 9, 45, 121
 problems with 6, 264
 RSS 221, 391
 Ruby 104, 272
 rule activation 331
 Rule Agent 384–385
 rule domains 394
 rule engine 328–331
 actions 330
 agenda 336
 characteristics 330
 common algorithms 330

improving software
 quality 329
 purpose is rendering a
 decision 354
 reduces complexity of
 code 328
 relationship to BRMS 332
 rule activation 330, 371
See also BRMS
 rule processing 245
 rule properties. *See* rule engine
 rule repository 390
 rule.properties 398
 RuleBase 338, 372, 398
 RuleBaseFactory 338
 RuleFlow. *See* Drools RuleFlow
 RuleFlowGroup 357
 rules engine 325

S

SaaS 9, 55, 286, 320, 329
 Salesforce 9, 96
 Salesforce.com 286
 salience 336
 SAML 63
 Sarbanes-Oxley Act 219
 SCA 44, 195
 accelerates path to SOA 201
 and the role of an ESB 266
 Apache Tuscany as
 implementation 64
 as a client 91, 196
 assembly model 64
 binding using custom
 WSDL 87
 bindings 87–93
 bindings schema 87
 callbacks 99–104, 108–113
 illustration 99
 support 75
 support and usage 99
 case study requirements 65
 client using SDO 117
 code annotations 95
 component
 architecture 53
 implementation 70
 implementation node 79
 schema 70
 type as an alternative to
 annotations 95
 type file example 95
 type XML file 95
 types 95–96

- SCA (*continued*)
- components 70–74
 - composite 53, 66–70
 - composite diagram
 - example 67
 - composite file example for Esper 248
 - composite XML Schema 66
 - composition example using properties 79
 - configuration of decision service 398
 - conversations 96–99
 - composite scope 97
 - conversation scope 97
 - request scope 97
 - stateless scope 97
 - creating Esper
 - components 246
 - creating protocol-neutral components 266
 - decomposing composite files for manageability 208
 - distinguishing between callbacks and conversations 99
 - domain 64, 118
 - example of test client used to submit remote service request 199
 - example of using conversational features 247
 - example of using reference 199
 - historical foundations 64
 - implementation 79–84
 - implementation.composite
 - example 83
 - implementation.java
 - element 79
 - injection options for properties 78
 - integrating with JBoss
 - jBPM 196–201
 - integration with Esper 246
 - Java annotations 71
 - Java conversation example 97
 - multi-protocol support 74
 - nodes and composite diagram 66
 - OASIS sponsorship 54
 - properties 53, 76–79
 - example 76
 - example alternative 77
 - using XML 76
 - with source XPATH 78
 - protocol and language
 - neutral 62
 - recursive composition 84
 - reference implementation
 - example 84
 - using web service 85
 - references 53, 64, 84–86
 - schema 84
 - relationship between WSDL and composite binding 89
 - relationship to OSGi 53
 - sca-contribution.xml 109
 - scripting implementation in composite.xml 106
 - scripting language
 - support 104–108
 - example 105
 - SDO's relationship 114
 - sequence diagram illustration
 - for decision service 403
 - service binding 74
 - service interface element 74
 - service interface for restricting exposed methods 75
 - service schema 74
 - service using
 - interface.wsdl 75
 - services 74
 - simplified overview 53
 - Spring 54
 - steps in setting up client for jBPM 196–201
 - tooling 65
 - using component type file for scripting languages 107
 - using
 - DomainManagerLauncher 113
 - using embedded container 108
 - using the promote attribute 69
 - using top-level service definition 68
 - using web service as reference interface 199
 - when to use callbacks 99
 - why it's so compelling to use with jBPM 196
 - SCA Tools 65
 - sca, composition 63
 - SCADomain 249
 - scripting languages 104, 263
 - SDO 53, 169, 195
 - advanced features 119–121
 - API 114
 - as a data graph 53
 - as an XML binding
 - technology 113
 - converting to XML 120
 - data graphs 119
 - disconnected datasets 114
 - example of marshalling XML for log4j 119
 - example used in decision services 400
 - example using raw XML 120
 - factory helper classes 119
 - for complex XML data structures 114
 - metadata 114
 - steps for unmarshalling XML into Java classes 120
 - steps for using with SCA 115–119
 - test client 117
 - usage 113–119
 - use with XML Schema 114
 - using to marshall object data into XML 119
 - when to use 198
 - working with generated classes 119
 - SDOClientMain 118
 - SDOUtil 119
 - security policies 315
 - SeeBeyond 254
 - SeeWhy 47
 - selection mediators 272
 - sense and respond 134
 - sequence 275
 - service 61
 - benefits 63
 - components and
 - compositions 62–64
 - open source evaluation 52–54
 - composite 13
 - composition 14–15
 - caevats for ESB 266
 - consumer 213
 - contract 10
 - detection 265
 - differences from a component 62
 - discrimination 261
 - fine vs. coarse grained 63
 - interface 10–11
 - lifecycle management 52
 - loose coupling 13–14, 74

- service (*continued*)
 - mediator 12
 - registry 15–16
 - scalability 108
 - stateful vs stateless 13
 - transparency 11
 - wrappers 23
 - service adapters 255
 - Service Data Objects. *See* SDO
 - service level agreements. *See* SLAs
 - service mediation 25
 - depiction 25, 55
 - open source evaluation 55–56
 - See also* web service mediation
 - Service Oriented Architecture. *See* SOA
 - service provider 213
 - service proxy 266
 - service virtualization 260, 265, 318, 390
 - service-enabling Esper 245–250
 - services 8, 23
 - as holy grail of SOA 62
 - asynchronous 99
 - components and composites 23–24
 - composite 14
 - conversational support using SCA 99
 - fine and coarse-grained 13
 - should not contain protocol-specific code 266
 - simple vs. primitive 14
 - stateless 96
 - See also* SOA
 - servlet 110, 113
 - SessionManagerImpl 247
 - setPayloadXML 279
 - sFTP 253
 - signal 193
 - signaling a token 201
 - Simple Object Access Protocol. *See* SOAP
 - SimpleURLRegistry 290
 - SLAs 167, 242, 273
 - and BPM 134
 - Smith, Howard 17, 126
 - SMTP 105, 271, 304
 - snapshots 384
 - SOA 7, 329–330
 - advent 7
 - and BPM 127
 - and the role of an ESB 253
 - as a fad 3
 - benefits 9
 - challenges introduced for managing and monitoring 218
 - core characteristics 10–16
 - disappointments 9
 - distributed environment 218
 - environment 8
 - facilitates event publication 220
 - Harvard Business Journal 4
 - inappropriate uses of an ESB 265, 268
 - introduces many more failure points 218
 - maturity model 25
 - more than SOAP 10
 - orchestrated business processes 8
 - relationship to business rules 329
 - SOAP web services 9–10
 - technology platform 16–25
 - SOAP 6, 36, 55, 74, 195, 252, 258, 270, 287
 - addressing 274
 - API 286
 - authentication 316
 - benefits for using with JMS 89
 - document style 6
 - example mock service 290
 - example request 290
 - fault 290
 - headers 260, 270, 305
 - messaging styles 6–7
 - over JMS 287
 - relationship to SOA 9–10
 - RPC/encoded 7
 - vs REST 10
 - XML 6
 - soapUI 111, 117, 250, 279, 287, 290, 292, 318
 - example depiction 403
 - for testing of decision services 403
 - Soar 330
 - socket 6
 - Software AG 44
 - Software as a Service. *See* SaaS
 - Sonic Software 41, 56, 252
 - sort 237
 - SourceForge 129
 - SPI 262
 - split 272
 - splitter 357
 - Spring 54, 65, 76, 84, 147, 249, 403
 - Spring-WS 10
 - SQL 227, 313
 - sql 245
 - Squid 56
 - SSL 262
 - state transitions 139
 - stateless 390
 - StatementManagerImpl 247
 - StatementSubscriber 246
 - stored procedures 245
 - stovepiped applications 31
 - streams. *See* Esper
 - StringUpdateLog 190
 - Stylus Studio 204
 - subprocesses. *See* JBoss jBPM, subscriber 226
 - Subversion 385
 - SuccessFactors 286
 - sum 237
 - Sun Microsystems 5
 - Sun OpenESB 252
 - super-state 182
 - superstate-enter 182
 - superstate-leave 182
 - superstates. *See* JBoss jBPM, superstates
 - swimlanes 161, 164
 - jBPM process flow depiction 164
 - when are they necessary? 164
 - See also* JBoss jBPM, swimlanes
 - switch mediator 277, 306
 - Synapse
 - architecture 45
 - service proxy configuration 280
 - See also* Apache Synapse
 - synchronous communications 99
-
- T**
- tagging 380
 - Talend 26
 - task
 - controllers 168
 - jBPM task variables 168
 - transitions 160
 - task assignment 161
 - task duedate attribute 161
 - task form 159
 - task node. *See* JBoss jBPM, task nodes
 - task priority 161

TaskControllerHandler 146, 169
 TaskInstance 174–176, 178
 TaskInstance.getPooledActors() method 175
 TaskMgmtInstance 177
 TaskMgmtSession 176
 task-node 158, 160
 tasks
 completing a task using jBPM API 177–178
 determining which task to close when using the jBPM API 178
 finding open tasks using jBPM API 174–175
 finding pooled tasks assigned to users using jBPM API 176
 finding tasks assigned to a user using jBPM API 176
 relation to BPM and SOA 158
 roles 164
 user management 161–165
 using the jBPM API 169–178
 See also JBoss jBPM
 Taylor, James 329, 390
 TCP 271
 temporal operators 241
 TestNG 403
 Thread.sleep 101
 throttle mediator 314
 ThrottleAssertion 319
 throttling 286
 throttling assertions using WS-Policy 319
 Tibco 254
 Tibco General Interface (GI) 169
 timer 138
 at 243
 interval 243
 jBPM configuration example 166
 within 242
 timers
 add to jBPM superstates 182
 in jBPM 165, 167
 times, only available in jBPM Enterprise 165
 Token 185
 tokens 177
 See also JBoss jBPM tokens
 Tomcat 334

Traffic Management 55
 transformation 255
 TransitionLog 190, 192
 transitions 138, 144
 transport switching 253, 287
 transports 253
 Treat 330
 triggers 273
 Tuscany SDO. *See* SDO

U

ubl.xsd 290
 ubl_order.wsdl 288
 UDDI 15, 48
 UDF. *See* Esper, user defined functions
 UML 164
 uniform resource locator. *See* URI
 Universal Business Language. *See* OASIS UBL
 Universal Description, Discovery and Integration. *See* UDDI
 UpdateListener 241
 updateRStream 234
 updateToken 203
 URI 260
 Username 317
 UsernameToken 271, 316

V

validate mediator 285, 290
 validation 258, 272
 VariableCreateLog 190
 VariableDeleteLog 190
 VariableInstance 155
 variables. *See also* JBoss jBPM, variables
 version control 385
 Version Rationalization 55
 VFS metadata 305
 Virtual File System. *See* Apache Commons VFS
 virtual machine 113
 Visio 126
 visualization 126
 Vitria 254
 VMWare 264
 volatile vs. non-volatile facts 331
 Von Halle, Barbara 35, 327

W

WADL 10
 wait states 356
 jBPM behavior 192
 WAR file 109
 web forms 159
 Web Service Definition Language. *See* WSDL
 web service mediation 25, 260, 286–298
 See also service mediation
 web service, versioning 318
 Web Services Security UsernameToken Profile 1.0 314
 web services, jBPM facade 195
 web.xml 109–110
 WebLogic 54
 WebMethods 254
 Web-Oriented Architecture 10
 WebSphere 54
 win, keepall() 237
 wire-tap 285
 pattern 304–308
 wiretap 259
 Woolf, Bobby 253, 259
 work items 357
 workflow 14, 127
 using Drools RuleFlow 356
 workflow management. *See* BPM
 working memory 328, 330
 case study fact objects 366
 how to populate 392
 inserting 354
 loading non-volatile facts using Digester 374
 non-volatile, volatile facts 331
 populating instance, or volatile, data 375
 querying facts 356, 376
 removing objects using retract 355, 377
 retracting facts 338
 updating 354
 updating fact objects 369, 371
 use in decision services 390
 using Digester to populate from XML 372
 using SDO to populate via web service 401
 WorkingMemory 339, 371–372, 375–376, 402
 WS-* 12

- WSO2 ESB 269
 - differences with Apache Synapse 269
 - WSO2 Registry 16
 - ATOM/RSS support 51
 - compared to Galaxy 51
 - elements 51
 - WS-Addressing 11–12, 44, 260, 264, 270, 272, 274, 279
 - use with WS-Security 316
 - WS-BPEL 14, 133, 136, 267
 - differences to BPM 32
 - WSDL 6, 24
 - and XML Schema 10
 - as a registry 15
 - best practices 210
 - creating simple 288
 - decision services 393–396
 - example of constructing manually 204
 - example using JMS transport 294
 - for Esper web services 250
 - interface 10
 - multiple service definitions 88
 - top-down vs. bottom-up design 393
 - using with SCA/Apache Tuscany 88
 - using wsdl import statement 395
 - WS-I 117
 - WSO2 319
 - Registry, introducing 51
 - role in Apache Synapse 269
 - See also* Apache Synapse
 - WS-Policy 44, 271, 315
 - example 318
 - extended for QoS 318
 - WS-ReliableMessaging 25, 44, 271
 - WS-Security 25, 44, 55–56, 262, 286, 314–317
 - example 317
 - set of standards 314
- X**
-
- X.509 271, 315
 - XAware 392
 - XEN 264
 - xis2.xml 292
 - XML 255
 - binding with SDO 121
 - composite definition 66
 - large payloads 13
 - marshalling into Java class example 120
 - namespace 109
 - preferred for ESB messaging 258
 - tracking changes to a document 114
 - transformation 287
 - views 392
 - XML Schema 51, 116
 - validation 287
 - XMLBeans 114
 - XMLDocument 120
 - XMLHelper 119–120
 - XOP 272
 - XPath 272
 - XPath expression 290, 306, 312–313
 - XPDL 136
 - XQuery 45, 104, 272
 - XSD2JavaGenerator 116, 197, 205
 - xsi, type attribute 395
 - XSLT 104, 272
 - mediator 285
 - stylesheet 314
 - transformation 261
 - xslt
 - depiction 296
 - mediator 291, 303