Index

Symbols

!= operator 27
== operator 27, 138
@serial tag 43
@serialData tag 43
@serialField tag 44

A

abstract remote, design pattern 162–164
AccessException 124
AccessException 57, 201, 243
Activatable 80, 117–119, 121–122, 161, 182, 185
exportObject method 114, 118, 132, 157
with socket factories 118, 136
extending 114, 117–118, 133, 157
getID method 122, 133
inactive method 126
register method 114, 121, 133, 252
remote object semantics of 80
unexportObject method 126, 248
see also activatable remote objects
activatable proxy, design pattern 157–158
activatable remote objects 82, 87, 113–134, 157–158, 228
and security manager 87
as singletons 161
building 123
clients of 129–130
codebase 115, 123
construction of 132
by activation system 114–115, 117–118
with socket factories 136
constructor required for 114, 117
exporting 118
with socket factories 136
initialization argument of 115
references to, are persistent 82, 113
registration of 114, 117, 119–123
run-time setup for 123–125
secure 212–213
simultaneous registration and export 132
unregistration of 122
writing of 117–119
see also Activatable
activatable servers
see activatable remote objects
ActivateFailedException 116, 130, 243, 250
activation 5, 32, 51, 61, 63, 82, 113–134, 200
critique of 131
exceptions in 129
not supported by RMI/IIOP 185
activation daemon see rmid
activation group 74, 114, 119, 124, 126, 128–129, 132
creation of 132
in Win32 129
registration of 114, 117, 120–121
activation system, as a registry 127–128
ActivationDesc 121, 132, 161, 253
ActivationException 128–129, 132, 243
ActivationGroup 116, 133, 164
activeObject method 133
createGroup method 121, 132
getSystem method 115, 128
inactiveGroup method 133
inactiveObject method 133
newInstance method 133
ActivationGroupDesc 120, 124, 133
CommandEnvironment static inner class 120
ActivationGroupID 120–121, 132–133, 253
ActivationID 115–116, 121, 133, 161, 254
activate method 115, 133
ActivationInstantiator 133
ActivationMonitor 133
ActivationSystem 115, 128, 133
getActivationDesc method 115, 133
getActivationGroupDesc method 133
registerGroup method 120, 133
<table>
<thead>
<tr>
<th>Method/Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>registerObject</td>
<td>method 121, 133</td>
</tr>
<tr>
<td>setActivationDesc</td>
<td>method 115, 133</td>
</tr>
<tr>
<td>setActivationGroupDesc</td>
<td>method 133</td>
</tr>
<tr>
<td>unregisterGroup</td>
<td>method 133</td>
</tr>
<tr>
<td>unregisterObject</td>
<td>method 133</td>
</tr>
<tr>
<td>Activator</td>
<td>133, 214, 257</td>
</tr>
<tr>
<td>activate</td>
<td>method 115</td>
</tr>
<tr>
<td>adapter, design pattern</td>
<td>148–150, 209</td>
</tr>
<tr>
<td>Serializable</td>
<td>149–150</td>
</tr>
<tr>
<td>agents in RMI</td>
<td>143–148</td>
</tr>
<tr>
<td>callback (immobile agent)</td>
<td>145</td>
</tr>
<tr>
<td>mobile</td>
<td>144</td>
</tr>
<tr>
<td>mobile server (call-forward)</td>
<td>147</td>
</tr>
<tr>
<td>AllPermission</td>
<td>94</td>
</tr>
<tr>
<td>AlreadyBoundException</td>
<td>58, 244</td>
</tr>
<tr>
<td>Applet</td>
<td>init method 88</td>
</tr>
<tr>
<td>start method 88</td>
<td></td>
</tr>
<tr>
<td>stop method 88</td>
<td></td>
</tr>
<tr>
<td>application firewalls</td>
<td>191, 199</td>
</tr>
<tr>
<td>application logging</td>
<td>227–228</td>
</tr>
<tr>
<td>architecture of RMI systems</td>
<td>4–5</td>
</tr>
<tr>
<td>argument-passing</td>
<td>15–16</td>
</tr>
<tr>
<td>“at least zero” semantics</td>
<td>217, 220, 222</td>
</tr>
<tr>
<td>“at most once” semantics</td>
<td>15–16, 217, 245–247</td>
</tr>
<tr>
<td>authentication</td>
<td>106, 173, 197, 205–206, 210, 213</td>
</tr>
<tr>
<td>Authenticator</td>
<td>106, 197</td>
</tr>
<tr>
<td>authorization</td>
<td>206, 210</td>
</tr>
<tr>
<td>Class</td>
<td>ClassCastException</td>
</tr>
<tr>
<td>ClassCastException</td>
<td>57, 243–244</td>
</tr>
<tr>
<td>ClassCastException</td>
<td>145–147, 159, 227</td>
</tr>
<tr>
<td>and firewalls</td>
<td>202</td>
</tr>
<tr>
<td>limitation</td>
<td>147</td>
</tr>
<tr>
<td>and GIOP proxy</td>
<td>199</td>
</tr>
<tr>
<td>and HTTP tunnelling, limitation</td>
<td>194, 198</td>
</tr>
<tr>
<td>and RMI Proxy</td>
<td>201</td>
</tr>
<tr>
<td>and SOCKS, limitation</td>
<td>193</td>
</tr>
<tr>
<td>as security risk</td>
<td>202</td>
</tr>
<tr>
<td>deadlock in</td>
<td>146</td>
</tr>
<tr>
<td>installation/distribution of</td>
<td>79</td>
</tr>
<tr>
<td>CGI script</td>
<td>195–199</td>
</tr>
<tr>
<td>cipher suites</td>
<td>209–210</td>
</tr>
<tr>
<td>Class</td>
<td>getDeclaredFields</td>
</tr>
<tr>
<td>get name</td>
<td>method 56, 121</td>
</tr>
<tr>
<td>getResource</td>
<td>method 104</td>
</tr>
<tr>
<td>getResourceAsStream</td>
<td>method 104</td>
</tr>
<tr>
<td>class versioning</td>
<td>31–35</td>
</tr>
<tr>
<td>alternative approaches to</td>
<td>39</td>
</tr>
<tr>
<td>automatic, limitations of</td>
<td>33</td>
</tr>
<tr>
<td>in RMI/IIOP</td>
<td>35, 185</td>
</tr>
<tr>
<td>by derivation</td>
<td>39–40</td>
</tr>
<tr>
<td>examples of</td>
<td>37</td>
</tr>
<tr>
<td>ClassCastException</td>
<td>26, 97, 101–102, 243–244, 247, 254</td>
</tr>
<tr>
<td>ClassCastException</td>
<td>CLASSPATH</td>
</tr>
<tr>
<td>and registry</td>
<td>244–245</td>
</tr>
<tr>
<td>client socket factories</td>
<td>51, 73, 75, 136–137, 194, 198–199, 235</td>
</tr>
<tr>
<td>and stubs</td>
<td>136</td>
</tr>
<tr>
<td>must be serializable</td>
<td>137</td>
</tr>
<tr>
<td>client sockets</td>
<td>51, 57, 75, 135–136, 194</td>
</tr>
<tr>
<td>sharing of</td>
<td>75, 231</td>
</tr>
<tr>
<td>client subject, security</td>
<td>212–213</td>
</tr>
<tr>
<td>client threads</td>
<td>73</td>
</tr>
<tr>
<td>client-dispatcher-server, design pattern</td>
<td>159</td>
</tr>
<tr>
<td>clients</td>
<td>5, 7, 4, 49–52</td>
</tr>
<tr>
<td>and security managers</td>
<td>87</td>
</tr>
<tr>
<td>downloading of</td>
<td>108</td>
</tr>
<tr>
<td>exceptions in</td>
<td>14, 17, 242</td>
</tr>
<tr>
<td>installation of</td>
<td>32, 103</td>
</tr>
<tr>
<td>of activatables</td>
<td>129–130</td>
</tr>
<tr>
<td>of portable remote objects</td>
<td>186</td>
</tr>
<tr>
<td>non-Java</td>
<td>187</td>
</tr>
</tbody>
</table>

**B**
- backlog, in listen 74
- BAD_PARAM exception (CORBA) 34
  - bind operation
    - of JNDI 170
    - of registry 55
  - bootstrapping 8, 53, 57, 108–109, 111, 115, 169, 218
    - via an RMI method 110
    - via MarshalledObject 111
    - via RMIClassLoader 109
- broadcast 66, 220, 222–223
- BufferedInputStream 149
- BufferedOutputStream 153
client-server applications 3
communications 215
debugging 229
design pattern 159
Cloneable 67–68, 71
cloning 67–69, 71, 119, 249
code mobility 39–40, 60–61, 79, 82,
86–111, 162, 244
and registry 101–102
and security 99
uses of 97–99
code source 89–90
codebase feature 32, 39, 59–60, 64, 86, 89,
95, 97–98, 108–109, 201, 239, 245, 253
and activation 114–115, 123
and security 87, 93, 99–100, 106, 208
deployment with 107
format of 96–97
in activation 115, 123
protocols for 105–106
servers 5, 99, 102, 104
setup 100
Common Data Representation (CDR) 179
Common Object Request Broker Architecture see CORBA
ConnectException
java.net 202–203, 245
java.rmi 245, 250, 254
ConnectIOException 245, 250
ContextFactory 172
CORBA 23, 34, 82, 169, 179–188,
199–200, 267
Object By Value, feature of 2.3 183
COS Naming service 5, 169, 173, 183–184
JNDI provider 173
credentials, in security 206
cryptography see also encryption 206
 cyclic redundancy check (CRC) 207

D
database operations 50
Datagram 215
datagrams 215–223
DatagramSocket 215, 218–219, 222
DataInput 29
DataInputStream 149
DataOutputStream 153
Date 28
deadlock, in callback pattern 146
debugging 87–88, 90, 128, 261
deployment
of RMI registry 59
of RMI systems 107
deserialization 21, 26–30, 41
design patterns 39, 46, 143, 148
activatable proxy 157–158
adapter 148–150, 209
callback 75, 79, 106, 125–126, 145–147,
159, 193–194, 198–199, 201–202, 227
deadlock in 146
client-dispatcher-server 159
client-server 159
proxy 8, 46, 152–159
remote factory 39, 161–162
remote proxy 156
session 160, 164
singleton 54, 122, 160–161
smart proxy 152
smart remote proxy 156
virtual proxy 157
DGC
clean method 226
DGC see distributed garbage collection
directory contexts (JNDI) 171
dispatchers 47, 78, 159, 164, 228
Distributed Computing Environment (DCE) 23
distributed garbage collection 15, 70,
225–227, 235, 239
limitations of
and network partitions 226
in distributed cycles 227
not supported by RMI/IIOP 185, 227

E
early versions of RMI 237–239
eavesdropping on RMI
conversations 207–208
efficiency of RMI 232–233
enclaves, firewall 108, 200–201
encryption 30, 106, 208–209
endpoint factories 212–213
Enterprise Edition see J2EE
Enterprise Java Beans 180
| **EOFException** | 30 |
| **Error** | 241, 247, 249 |
| event notification (JNDI) | 171 |
| evolution | 32, 34 |
| of classes | 163, 251, 253 |
| of remote interfaces | 253 |
| of remote objects | 253 |
| **Exception** | 11, 247, 255 |
| **printStackTrace** method | 255 |
| exceptions | 170, 174–175 |
| in activation | 129 |
| listing of | 243–254 |
| in serialization | 22, 25, 29–30, 35 |
| **ExecOptionPermission** | 124–125 |
| **ExecPermission** | 124–125 |
| execution security | 85–94 |
| exiting when idle | 75, 77, 126, 134 |
| **ExportException** | 246 |
| exporting remote objects | 69 |
| eXternal Data Representation (XDR) | 23 |
| **Externalizable** | 22, 28, 30–31, 34–35, 38, 137, 237 |
| changing to **Serializable** | 33 |
| constructor required for | 31 |
| faster than **Serializable** | 31 |
| **readExternal** method | 30–31, 237 |
| **writeExternal** method | 30–31, 43, 237 |

| **F** |
| federation, of naming services (JNDI) | 171 |
| file system provider (JNDI) | 176 |
| file transfer protocol (FTP) | 105 |
| file: protocol | 105 |
| FileInputStream | 150 |
| FileOutputStream | 154 |
| firewalls | 74, 105, 107–108, 139, 147, 189–203, 258 |
| vendor-specific issues | 202 |
| first principles of RMI | 6–7 |
| flat namespace, of registry | 55, 63, 169, 176 |
| foundation classes of servers | 79–81 |

| **G** |
| garbage collection | 70, 126–127 |
| distributed see distributed garbage collection |
| General Inter-ORB protocol see GIOP |
| GIOP | 179 |
| proxies | 199–200 |

| **H** |
| Hashtable | and de-serialization 27 |
| home pages | for this book 268 |
| Java RMI | 267 |
| Java Software | 267 |
| HTML (Hypertext markup language) | 191 |
| HTTP | 103 |
| lightweight class server | 104 |
| system properties for | 192 |
| tunnelling | 193–194, 198–199, 209 |
| limitations of | 194 |
| http.nonProxyHosts | 257 |
| http.proxyHost | 192, 194–195, 199, 257 |
| http.proxyPort | 192, 195, 199, 257 |
| HTTPS (secure hypertext transfer protocol) | 106 |
| http.proxyHost | 197 |
| https.proxyHost | 106 |
| https.proxyPort | 106 |
| HttpServlet | 197 |
| hypertext transfer protocol see HTTP |

| **I** |
| idempotence | 17, 50, 167 |
| identity, of party to conversation | 205 |
| IDL | 23, 34, 180, 182–183, 185–186, 188 |
| idle, exiting when | 75, 77, 126, 134 |
| idle-time activity | 75 |
| IIOP | 82, 173, 179–188, 199, 267 |
| InetAddress | 140 |
| InitialContext | 183 |
| initialization argument, of activatable | 115 |
| remote objects | 115 |
| InputStream | 150 |
| **read** method | 149 |
installation
  of callbacks 79
  of clients 32, 103
  of RMI registry 59
  of RMI systems 107
  of servers 32
integrity, of message 206, 209
interface definition language see IDL
Internet 189–190, 193, 200, 203, 219
Internet Inter-ORB protocol see IIOP
InterruptedException 51
introspection 36, 43
InvalidClassException 35
IOException 25, 150, 154–155, 246, 254
IP 215, 219

J

J2EE 104–106, 180, 210
Java 2 Enterprise Edition see J2EE
Java Naming and Directory Interface see JNDI
Java remote method protocol see JRMP
Java Secure Sockets Extension see JSSE
java.content.handler.pkgs 106
java.naming.factory.initial 172, 174
  for COS Naming provider 173
  for LDAP provider 173
  for RMI registry provider 172
java.naming.provider.url 172–173, 175
java.rmi.activation.activator.class 257
java.rmi.activation.port 258
java.rmi.activation.security.class 257
java.rmi.activation.security.codebase 257
java.rmi.dgc.leaseValue 236, 258
java.rmi.server.codebase 60, 96, 101–103, 109–110, 244, 258–259
java.rmi.server.disableHttp 198–199, 258
java.rmi.server.hostname 142, 236, 258–259
java.rmi.server.logCalls 258
java.rmi.server.randomIDs 259
java.rmi.server.useCodebaseOnly 103, 259
java.rmi.server.useLocalHostname 259
java.security.debug 88, 259
java.security.manager 85, 88, 259
java.security.policy 88, 124, 252, 260
javadoe tags, for serialization 43
java-rmi CGI script 196
JavaSpaces 167
JDK versions, interoperability of 238–239
Jini 176–177
  Discovery Service 176–177
  Lookup Service 176–177
JNDI 169–177, 183, 225
  attribute 171
  context 170
  directory context 171
  event notification 171
  exceptions 170, 174–175
  federation, of naming services 171
  file system provider 176
  initial context 183
  initial factory property 172
jndi.properties file 175
LDAP provider 173
  listener 171
  operations 170
  provider 169–171, 176–177
  URL property 172
rename operation 177
RMI registry provider 172, 183
service provider interface (SPI) 171
setup 175
sub-context 170
jndi.properties file 175
JRMP 5, 199
JSSE 106, 138, 209

L

 latency 51
LDAP 169, 171, 173
  JNDI provider 173
Lease 226
leaseValue 226, 235
Lightweight directory access protocol see LDAP
list operation
  of JNDI 170
  of registry 55
listen backlog 74
load-balancing 122, 159
<table>
<thead>
<tr>
<th>M–P</th>
<th>Index</th>
<th>java.rmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>local area network (LAN) 189</td>
<td>NoClassDefFoundError 97, 243, 247</td>
<td></td>
</tr>
<tr>
<td>LocateRegistry 58</td>
<td>NoRouteToHostException 194, 199, 202–203</td>
<td></td>
</tr>
<tr>
<td>createRegistry method 59, 61, 74</td>
<td>NoSuchObjectException 70, 77, 125, 226, 247, 250</td>
<td></td>
</tr>
<tr>
<td>getRegistry method 57</td>
<td>NotBoundException 58, 174, 248</td>
<td></td>
</tr>
<tr>
<td>location transparency 159</td>
<td>NotSerializableException 22, 25–26, 28</td>
<td></td>
</tr>
<tr>
<td>logCalls 228</td>
<td>Novell NetWare 170</td>
<td></td>
</tr>
<tr>
<td>logging 80, 128, 227–229, 261–263</td>
<td>Network Directory Service 170</td>
<td></td>
</tr>
<tr>
<td>LogStream 228, 258</td>
<td>lookup operation of JNDI 170</td>
<td></td>
</tr>
<tr>
<td>of registry 55</td>
<td>of registry 55</td>
<td></td>
</tr>
</tbody>
</table>

**M**

main method 11, 88
MalformedURLException 58, 174
MarshalException 22, 246, 250–251
MarshalledObject 8, 31, 53, 59, 109, 111, 115, 121
bootstrap 53, 111
get method 31, 111
marshalling 18, 21, 46, 92, 96, 213, 218, 233, 246, 250
message digest 207
mobile agents 144
mobile code see code mobility
mobile servers 147
multicast 66, 219–222
MulticastSocket 219, 222
multi-homing, of network hosts 140–142
multiplexing protocol, of RMI 202

**N**

name collisions 255
names, in registry, and URLs 58
namespace, flat, of registry 55, 63, 169, 176
Naming 56–58, 200, 242–243, 248, 253
bind method 57–58, 101, 243–244
list method 58
lookup method 46, 58, 102, 244, 248
rebind method 57–58, 243
unbind method 57–58, 243, 248
naming services 53–63, 169–176, 186
and RMI/IIOP 184
federation of 171
NamingException 174–175
network analysers (sniffers) 207

276

Object 26, 28, 33
clone method 7–8, 67
equals method 7–8, 27, 73, 75, 137–138, 140
hashCode method 7–8
toString method 7–8
object graphs 23–27, 32
and portable remote objects 185
received and returned by distinct RMI calls are distinct 27
ObjectInputStream 149
and RMI/IIOP 35
defaultReadObject method 29–30, 37, 237
defaultWriteObject method 37
readObject method 26
object-oriented databases 54
ObjectOutputStream 153
and RMI/IIOP 34
annotateClass method 96
defaultWriteObject method 29–30, 42, 237
writeObject method 25
ObjectStreamField 43–44
OperationNotSupportedException 170
OptionalDataException 29
OutOfMemoryError 237, 241
OutputStream 153, 155–156

P

partial failure, in RMI 19, 50–51
peer-to-peer communications 160, 215
performance of RMI 230–237
permissions 51–52, 85–94, 99–100, 249
given to users 205
in RMI registry 243
in RMI, table of 91
in rmid 124
of client and server 205–214
to create server sockets 252
permissions granted to users 205
in RMI registry 243
in RMI, table of 91
in rmid 124
of client and server 205–214
to create server sockets 252
persistence in registry 63, 176
implementing 63, 122
of references to activatable remote objects 82, 113
point-to-point communications 66
policy files 86, 88–90, 94, 99, 120, 124, 252, 260
of rmid 123–124
policys 86, 88
polymorphism 40, 97–98, 143, 162–163
portable remote objects 82, 180, 185–188, 227
and class versioning 185
and COS Naming service 173
and IIOP 180
and naming services 184, 186
and object graphs 185
and rmi 180, 182
cannot extend RemoteServer 185
clients of 186
defined 82
differences from unicast and activatable 180
dual-protocol (JRMP and IIOP) 183
exporting 182
extending other classes 182
extending PortableRemoteObject 181
extending RemoteObject 181
non-Java clients of 187
non-Java implementations of 186
port numbers, not supported by 180
remote interfaces implemented by 185
casting of 185–186
require Objects By Value feature of CORBA 2.3 183
restrictions of 183–185
socket factories, not supported by 180
stubs and ties 182
Unreferenced, not supported by 185
writing of 181
see also PortableRemoteObject
PortableRemoteObject 82, 180, 183
construction of 180
exportObject method 180, 182–183
extending 181
narrow method 185
unexportObject method 180
see also portable remote objects
principal, in security 206
PrintStream 228
privacy, of conversation 207
privileged blocks 90, 93
PrivilegedAction
run method 214
Properties 120
protection domains 89–90, 94
providers, JNDI 172–173, 176
proxy, application (in firewall) 191
proxy, design pattern 8, 46, 152–159
activatable proxy 157–158
remote proxy 156
smart proxy 152
smart remote proxy 156
virtual proxy 157
proxy, GIOP 199
proxy, HTTP 107–108, 191, 195–197
proxy, SOCKS 192–193
R
readObject method, in serializable
class 29–30, 33–34, 37, 237
readResolve method, in serializable
class 33, 41
rebind operation
of JNDI 170
of registry 55
Registry 57–58, 242–243, 248, 253
bind method 57–58, 101, 244
list method 57–58
lookup method 57–58, 102
rebind method 57
unbind method 57–58
registration 4–5, 8–9, 52–63, 74, 83, 97, 113, 115, 119, 121–123, 136, 158, 160, 162, 180, 184, 186, 225, 227, 236, 242
activation system as a 127–128
alternatives to 63
and CLASSPATH 244–245
and code mobility 101–102
Remote

and persistence 63
as UDP service 218
configurations 60–61
flat namespace of 55, 63, 169, 176
hierarchical 176
JNDI provider for 172
persistence of 176
servers bound in 127, 161
stale 247
Sun debugging utility for 63
Remote 7, 9, 45, 54, 121, 185
remote exceptions 8, 14, 17, 49–50, 57, 80, 158, 220
listing of 243–255
remote factory, design pattern 39, 161–162
remote failure 49–50, 254
remote input stream 149, 153
remote interfaces 4, 6–7, 14–15, 45–47, 49, 55–56, 65, 71, 77, 114, 131, 144, 150, 156
and abstract remote pattern 162
and ClassCastException 244
and NoClassDefFoundError 247
and smart proxy pattern 152
casting of 244
conditions satisfied by 7, 45, 155
design of 155, 248
evolution of 163, 251, 253
implementing methods in 66, 70–71
in RMI/IIOP 185
and case of method names 185
and IDL mangled names 185
casting of 185–186
generating IDL from 188
inheritance of methods 185
origin of 183
internal, in activation 133
remote methods 45–46
and idempotence 17
and latency 51
and object graphs 27
and partial failure 50
and retrying 17
and serialization 21
and UnexpectedException 253
argument-passing 14
as bootstrap 110
concurrency of 73
debugging 229
defined 7
design of 15, 17
exception-passing 14
implementing 70–72
in activatable 115, 129
obtaining the client host of 80
overhead of 216, 233
protocol (JRMP) 5
remote exceptions in 241
result-passing 14
semantics of 14–15, 17
syntax of 13
remote objects 4, 6–7, 18–19, 45, 49, 65, 144, 160, 190, 194, 211, 213
activatable 82, 115, 157
and connection sharing 75
and DGC 15, 225
and latency 51
and marshalling 45–46
and method parameters 15
and mobile server pattern 147
and object graphs 32, 46, 185
and port sharing 73–74
and registry 53
and serialization 81
and ServerSocket 136
and socket factories 74, 136
and stubs 7
and threads 73
and Unreferenced 75, 125
as callbacks 145–146
as EJBs 180
as factories 161
as local objects 7
as MarshalledObjects 59
broadcast 222
call stack in 255
casting of 244
cloning of 67
conditions satisfied by 7, 15, 65
declared as RMI parameter or result 244
defined 6
evolution of 253
exiting when idle 76–77, 126
exporting 69, 72, 246, 252, 258
local garbage collection of 70
marshalling of 14
multicast 219
non-static inner class of 244
passed by reference 16
portable (IIOP) 82
properties of 6
secure identifiers for 259
semantics of 7, 14, 65–69, 79, 81, 119
stale references to 247
thousands of 236, 238
unexporting 70
forcibly 70
unicast 66, 82
and TCP/IP 66
unmarshalling of 254
remote output stream 153, 155
remote proxy, design pattern 156
RemoteCall 251
getResultStream method 251
not throwing 71
RemoteObject 65, 67, 69, 71, 79–81, 117, 119, 181, 185, 248
equals method 79–80
extending 68
getRef method 80
hashCode method 79–80
remote object semantics of 68, 80
toString method 79–80
RemoteRef 115–116, 218
extending 68
getClientHost method 65, 80, 251
getLog method 65, 80, 228, 258
has no abstract methods 80
only exports static methods 68
remote object semantics of 80
setLog method 80
rename operation, of JNDI 170
resource conservation, in RMI 231
ResourceBundle 104
RMI early versions of 237–239
RMI CGI script 195
RMI multiplexing protocol 202
RMI registry port 190
RMI Security Extension 211–214
RMI/IIOP 34, 179–188
see also portable remote objects
RMIClassLoader 109–110
bootstrap 109
loadClass method 95, 101, 109–110
RMIClientSocketFactory 57, 136, 209, 214, 218
createSocket method 136
rmid 63, 115, 123–129, 190, 228, 237, 243, 258, 261
debugging 128
policy class 124
policy file 123–124
RMICgiToCgisocketfactory 140
RMICgiToXXSsocketfactory 198
RMISecurityException 249
RMISeCURITYMANAGER 99, 249, 257
RMISeCURITYMANAGER 236, 209, 214, 218
createServerSocket method 136
RMISeCURITYMANAGER 238
createSocket method 236, 238
getDefaultFactory method 139, 198
setSocketFactory method 92, 135
RMISeCURITYMANAGER utility 229
rollout 39–40, 98, 108, 162
Runnable 108, 111
run method 88, 108, 111
RunTime 160
RunTimeError 124, 155, 241, 251
RunTimePermission
setSecurityManager target 86
S
scalability, of RMI 234
secure client 214
secure code sections 214
secure hypertext transfer protocol see HTTPS
secure impersonation 213
secure remote calls 214
secure servers 212–214
activatable 213
secure sockets layer see SSL
secure stub 212, 214
Security
doConstrained method 214
security
authentication 205
authorization 206
credentials 206
cryptography 206
identity, of party to conversation 205
in SSL 209
integrity, of message 206, 209
message digest 207
of conversation 205–214
of execution 85–94
principal 206
subject 206
security managers 51, 85–86, 88, 252, 257, 259
and activation 87, 120
and applets 86
and bootstrapping 109
and mobile code 87, 99
and non-activatable remote objects 87
in JDK 1.1 86, 249
SecurityException 51, 85–86, 88, 124, 249, 259
where to catch 88
SecurityManager 85, 99
semantics 13
"at least zero" 217, 220, 222
"at most once" 15–16, 217, 245–247
declared 13
of remote method invocation 14
of remote objects 14
as RMI parameter or result 14
serial version unique identifier see SUID
Serializable 22, 27, 29–30, 137, 233, 237
and RMI/IIOP 35
changing to Externalizable 33
constructor required for 28
declarer no methods 27
slower than Externalizable 31
serialization 4, 21–44, 67–69, 81, 92, 96, 98, 100, 119–121, 143, 157, 179, 185, 207, 233
advanced facilities in 41
and javadoc 43
and remote objects 81
class versioning in, automatic 32, 34
class versioning required for 28
customized 29
default 29–30
improving the performance of 237
overheads of 233
via Externalizable 30–31
see also class versioning
serialPersistentFields mechanism 33, 43
serialVersionUID 22, 36, 233, 237
server socket factories 73–74, 136, 246
for multi-homing 140
server sockets 135–136, 231
sharing of 73, 231
server threads 73, 138, 231
ServerCloneException 249
ServerError 247, 249
ServerException 245, 250
ServerNotActiveException 80, 251
ServerRuntimeException 251
servers 5
installation of 32
stateless 160
see also activatable remote objects;
HTTP servers; mobile servers; portable
remote objects; registry; remote
objects; secure servers; stateless ser-
vers; unicast remote objects
ServerSocket 72, 74, 93, 135–136, 140, 209, 212, 218, 221, 246
accept method 93, 234
and permissions 252
construction of 140–141
bindAddr parameter 141
in multi-homing 140
ServerSocketFactory 246
servlet handler 197
session, design pattern 160, 164
sessions 54, 74, 76
SSL 210
sharing
of client sockets 75, 231
of server sockets 73, 231
singleton, design pattern 54, 122, 160–161
SkeletonMismatchException 250–251
SkeletonNotFoundException 251, 255
smart proxy, design pattern 152
smart remote proxy, design pattern 156
sniffers (network analysers) 207
Socket 93, 135–136, 193–194, 198, 209, 212, 216, 218, 221, 238
setSocketImplFactory method 92
setSoTimeout method 236, 238
socket factories 51, 57, 67, 69, 81, 91, 118, 135–142, 236
client 51, 73, 75, 136–137, 194, 198–199, 235
must be serializable 137
default, in RMI 198, 263
equality of 73–75, 137–138
for datagrams 218, 221
280
for multi-homing 140
for registry 59, 136
for Unix domain sockets 230
in Java 2 136
in JDK 1.1 135
in JDK 1.2.2 238
in RMI, critique of 142
not supported by RMI/IIOP 180, 185
server 73–74, 136–137, 246
SSL 139, 209
uses of 139
sockets see client sockets; server sockets;
Unix domain sockets
SocketSecurityException 252
SOCKS proxy 192–193
socksProxyHost 193, 260
socksProxyPort 260
SSL 139, 209–210
SSLSocket 209
SSLSocketFactory 209
SSLSSession 164
socket 209
SSLSessionFactory 138, 209
stateless servers 160
static 22, 26, 43
String 33, 121, 185
StubNotFoundException 250, 252, 255
stubs 7–8, 16, 46, 53–54, 80, 133, 157, 211,
223, 258
and client socket factories 136
and remote objects 7
generation of 46, 77, 81
of collectable 53, 63, 115, 121, 123–124,
133, 157, 161, 180, 243
of datagram 218
of multicast 222
of portable remote object 182
of registry 57
of unicast 63, 66
properties of 8
protocols 47
stale 247
sub-context, in JNDI 170
Subject 213
doAs method 213
subject
client 212–213
in security 206
SUID 35–37
defining 36
obtaining 37
sun.rmi.activation.execPolicy 124, 260
sun.rmi.activation.execTimeOut 237,
260
sun.rmi.activation.snapshotInterval 260
sun.rmi.dgc.checkInterval 261
sun.rmi.dgc.cleanInterval 261
sun.rmi.dgc.client.gcInterval 261
sun.rmi.dgc.logLevel 261
sun.rmi.dgc.server.gcInterval 261
sun.rmi.loader.logLevel 261
sun.rmi.log.debug 261
sun.rmi.rmid.maxStartGroup 237, 262
sun.rmi.server.activation.debugExec 128,
262
sun.rmi.server.exceptionTrace 228, 262
sun.rmi.server.logLevel 262
sun.rmi.transport.connectionTimeout 235,
236, 262
sun.rmi.transport.logLevel 262
sun.rmi.transport.proxy.connectTimeout
235, 236
sun.rmi.transport.proxy.logLevel 263
sun.rmi.transport.tcp.connectionTimeout
236
sun.rmi.transport.tcp.logLevel 263
sun.rmi.transport.tcp.readTimeout 235,
236, 263
synchronized 71
syntax
defined 13
do codebase URL 96–97
do local method invocation 13
do RMI 5, 13
do wildcard, in policy file 125
System
err member 229
gc method 126, 237
runFinalization method 126
system properties 257–263
and activation 124, 128
and applets 86
for HTTP proxy configuration 192
for JNDI 172–173, 175
for RMI DGC 226
for RMI logging 228
for SOCKS configuration 193
in RMI specification 257–260

281
in Sun implementation 260–263
security related 87–88

T

backlogs in listen 75
testing
of RMI in a single machine 230
of security policies 94
Thread
run method 88
ThreadGroup
uncaughtException method 88
threads 72
and remote methods 73, 231
client 73
collection of 231–232
creation overhead 74
deadlocks, in callback 146
killing of 237, 241
listening 74
server 73, 138, 231
reducing usage of 235–236
thread-safety 73
tie, CORBA 182
timeouts 20, 51, 135, 199, 218, 229, 231, 236, 238, 260, 262–263
tuning of 235
TLS see transport layer security
transient 22, 26, 43, 255
transport firewalls 190–191
transport layer security 209
tuning techniques 235–237

U

UDP 190, 215–219, 222
unbind operation
of JNDI 170
of registry 55
UnexpectedException 253
unexporting remote objects 70
unicast remote objects 65–82, 113, 117, 125, 179, 215, 228
and TCP/IP 66
building of 77
changing to activatable 157
clients of 129
exporting 69, 118
with socket factories 136
extending other classes 69
extending RemoteObject 68
extending RemoteServer 68
extending UnicastRemoteObject 66
references to, are transient 82, 113
secure 212
unexporting 70, 248
forcibly 70
writing of 66
see also UnicastRemoteObject
unicast server see unicast remote objects
UnicastRemoteObject 9, 63, 65–66, 69, 71, 80, 180, 185
auto-exported on de-serialization 81
cloning of 67, 249
collection of 67
with socket factories 136
exported on construction 69
exportObject method 69, 113, 127, 157, 183
with socket factories 136
extending 67, 81, 113, 157
not supported in RMI/IIOP 182–183
has only transient data? 81
port 81
remote object semantics of 67, 80
serialization of 67–68, 81
socket factory settings 81
unexportObject method 70, 248
force parameter 70
see also unicast remote objects
uniform resource locator see URL
Unix domain sockets 230
UnknownGroupException 130, 243, 253
UnknownHostException 130, 243, 253
UnknownObjectException 130, 243, 254
UnmarshalException 17, 22, 51,
101–102, 250–251, 254
unmarshalling 18, 21, 92, 96, 102, 213,
218, 233, 250, 254, 261
Unreferenced 75–76, 125, 226
RMI/IIOP, not supported in 185
unreferenced method 75–77, 125–126,
185, 226, 237, 247
may execute prematurely 226
URL 56–58, 89, 95–97, 101–102, 105–106,
   109, 111, 115, 121, 167, 172, 174–175,
   192, 195–196, 209, 252, 257–258, 260
   and names in the registry 58
User Datagram Protocol (UDP) 190,
   215–219, 222

V
virtual proxy, design pattern 157

W
wide area network (WAN) 189
wild-card syntax 125
Win32, activation groups in 129
writeObject method
   in serializable class 34
writeObject method, in serializable
class 29–30, 33, 43, 237
   and RMI/IIOP 34–35
writeReplace method, in serializable
class 33, 41–42

X
X.509 certificate chains 205
XDR 23