

Index

A

Abort() method, 664

abstract base classes, 20-21. See also COM [NPN]

ACE (Access Control Entries), 340

ACID transactions, 152

ACL (access control lists), claim-based authorization versus, 340

Action parameter (OperationContract attributes), 360, 516

Activate() method, streamed transfer mode, 532-534

activities (Windows Workflow Foundation)

ACID transactions, 152

binding, 139-140

CAG (ConditionedActivityGroup), 184

compensation, 152-153

custom activities

adding properties, 137

basic activities, 136-140

composite activities, 141

custom composite activities, 142

custom root activities, 165

DependencyProperty property, 138-140

XAML properties, 137

Delay, state machine workflows, 164

design behavior, 150-151

error handling, 134

- EventDrivenActivity, state machine workflows, 164
- HandleExternalEvent, state machine workflows, 164
- IfElse, sequential workflows, 155
- initializing, 134
- InvokeWebService, 190-191
- lifecycle of, 134
- out of the box activities, 135-136
- Parallel, sequential workflows, 157-163
- PlaceOrder activities, 134
- promoting properties, 140
- SetState, state machine workflows, 164
- tracing, services, 659-661
- validation, 150-151
- workflow communication host configuration, 145-149
- workflow communication interface, 143-145
- Add Web Reference dialog (Windows Workflow Foundation), consuming services, 190**
- AddItem message, service contracts, 552**
- addresses, 26-28**
 - absolute addresses as endpoints, 641
 - base addresses, specifying, 641
 - DerivativesCalculator example, 42-48
 - endpoint addresses, specifying, 641
 - PGM, 518. *See also* MSMQ PGM
 - purpose of, 77
 - Windows Communication Foundation, 641-642
- adopting Windows Communication Foundation**
 - delegating maintenance, 640-641
 - exemplars, 639
 - existing applications
 - incorporating, 640
 - rebuilding, 639
- AJAX (Asynchronous JavaScript and XML), 667-668**
- Announce message, service contracts, 552**
- anonymous identities, 260**
- anticipating exceptions, 664-665**
- API layers (frameworks), designing public layers, 647**
- Approval activity, 141**
- .asmx extension, 66**
- ASP.NET 2.0, enabling in CardSpace (Windows), 296-297**
- assertions (policies), 632**
- attaching custom behaviors to operations/endpoints, 425-427, 449-450**
- auditing security events, 661**
- authentication**
 - CardSpace (Windows), 280-285
 - COM+ integration, 386
 - FFIEC (Federal Financial Institutions Examination Council), 255
 - identity
 - Identity Metasystem, 259-273, 287
 - Information Cards, 266-273
 - Laws of Identity, 257-259
 - regulatory compliance, 256
 - role of, 253-255
 - user-centric identity management, 259
 - Kerberos authentication method, 280
 - Password (Microsoft), 257
 - passwords, 255
 - PKI (Public Key Infrastructure), 257
 - SSO (single sign on), 256, 286
- authorization. *See also* XSI [NPN]**
 - authorization domains, 339
 - claim-based versus role-based, 339-340
 - claim sets, 339, 362
 - COM+ integration, 386

- context (claim sets), 339, 362
 - FFIEC (Federal Financial Institutions Examination Council), 255
 - identity
 - Identity Metasystem, 259-273, 287
 - Information Cards, 266-273
 - Laws of Identity, 257-259
 - regulatory compliance, 256
 - role of, 253-255
 - user-centric identity management, 259
 - Password (Microsoft), 257
 - passwords, 255
 - PKI (Public Key Infrastructure), 257
 - resource access, Identity (Windows), 343-350
 - security
 - configuring services to use role providers, 240-241
 - service configuration, 243-245
 - ServiceAuthorizationManager implementation, 246-247
 - security, 239, 242, 248
 - SSO (single sign on), 256, 286
 - Authorization Manager**
 - Windows Server 2003, authorizing resource access, 344-350
 - XSI, claim-based authorization, 350-351, 354-357
 - Axis, WSDL2Java, 63-64**
- ## B
- backwards-compatible changes (versioning), 624**
 - base addresses, 48. See also addresses**
 - BasicHttpBinding, 46-48, 414, 523, 526, 643**
 - behavioral contracts, 652**
 - behaviors (custom)**
 - declaring, 421-426
 - dispatchers, 420
 - informing Windows Communication Foundation of, 426-435, 450
 - instance context provider, 445-446
 - instance provider, 446-447
 - message formatter, 440-442
 - message inspector, 442-444
 - operation invokers, 447
 - operation selector, 436-437
 - operations/endpoints, attaching to, 425-427, 449-450
 - parameter inspector, 438-440
 - usage examples, 451
 - WSDL export extension, 448-450
 - Bind dialog (Windows Workflow Foundation), 140**
 - binding, 28**
 - activities (Windows Workflow Foundation), 139-140
 - DerivativesCalculator example, 42-48
 - Peer Channel, 540-543
 - WSDL, 26
 - binding elements, 46**
 - custom binding elements
 - applying through configuration, 476-480
 - inbound communication support, 470-476
 - outbound communication support, 463-470
 - understanding starting points, 461-462
 - inbound communication, 455
 - inbound communication, 456
 - outbound communication, 454
 - BindingContext property, 459**

bindings

- activating/deactivating binding features, 646
- changing, versioning, 630
- custom message inspectors, 646
- custom operation selectors, 646
- custom peer resolver services, 645
- HTTP bindings, 644, 670
- identifying services with contended resources, 643
- predefined bindings, 643-645
- purpose of, 77
- queueing service requests, 643
- REST POX service endpoints, 572-573

**buffered transfer mode, 522. See also
streamed transfer mode**

BufferedDataSubscriptionManager class, 534

BufferedSubscriptionManager class, 532-535

BuildChannelFactory<T> method, 469

BuildChannelListener<T> method, 476

C

C++. See also COM [NPN]

- abstract base classes, 20-21
- encapsulation, 19
- interfaces, defining, 20-21
- messages, defining, 20-21

CAG (ConditionedActivityGroup) activity, 184

CalculateDerivative() operation, 78

callback contracts, 506-512

CallExternalMethod activity, workflow communication, 143, 147-148

CanBuildChannelFactory<T> method, 458-459, 464

CanBuildChannelListener<T> method, 458

CanBuildChannelListener<T> method, 459

CardSpace (Windows), 266, 286-289

- adding/installing cards, 268-270
 - architecture of, 273-276
 - ASPNET 2.0, enabling, 296-297
 - authentication methods, 280-285
 - card metadata, 273
 - catching exceptions, 313-314
 - Choose a Card dialog, 279
 - editing cards, 272
 - GetToken() method, 275
 - HTTP configuration, 300-301
 - IIS (Internet Information Services), enabling, 296-298
 - Information Cards
 - adding to browser applications, 317-328
 - adding to WCF applications, 301-313
 - building STS (Security Token Services), 332-334
 - creating Managed Cards, 328-332
 - processing security tokens, 314-316
 - Introduction dialog, 276-277
 - IssuedToken policy assertion, 275
 - Metadata Resolver, 316-317
 - mutual authentication dialog, 307
 - passwords, 280
 - preview dialog, 268
 - protocols, 278
 - security tokens, 273, 281
 - usage example, 295
 - X.509 certificates, 297-299
- centralized lifecycle management, 632-634**
- certificate store (CardSpace), importing
X.509 certificates to, 297-298**

certificates, transport security

- determining trustworthiness, 219-221
- identifying server provided certificates, 221-222
- installing, 219-221
- removing SSL configurations, 250
- specifying in SSL exchanges, 221-222
- uninstalling, 249

Channel Layer, 25, 77**channel listeners, custom transports, 493-496****ChannelFactoryBase<IRequestChannel>, 466****ChannelFactory<T> generic, MSMQ PGM, 517****ChannelListenerBase<IReplyChannel>, 472****channels (custom). See also Peer Channel**

- applying through configuration, 476-480
- binding elements, 454-456
- declaring shapes, 457
- inbound communication support, 470-476
- matching contracts to, 458-460
- outbound communication support, 463-470
- session support, 458
- shapes of, 456
- shapes of, 457
- state machines, 460
- understanding starting points, 461-462

Choose a Card dialog (CardSpace), 279**claim sets (security)**

- authorization context, 339, 362
- description of, 338
- evaluation context, 339, 362
- STS, 341
- WS-Trust, 341

claims (security)

- claim-based authorization, 378-379
 - ACL versus, 340
 - adoption strategies, 341
 - authorizing resource access, 350-351, 354-357
 - role-based authorization versus, 339-340
- description of, 338
- federated security versus, 380
- Identity Metasystem, 260
- normalization, 342
- rights of, 338
- SAML, 341
- types of, 338
- values of, 338

Claims section (.ini files), Managed Cards, 331**class file format, 21****class libraries, building service types, 656****client message formatters, 422****Client project, streamed transfer mode, 522****Client.csproj console application, 305-306****ClientBase<JT>, DerivativesCalculator example, 60-61****clients**

- AJAX, 667-668
- anticipating exceptions, 664-665
- asynchronous pattern for operation contracts, 662-664
- COM+/Windows Communication Foundation integration, 400-401
- DerivativesCalculator example, 53-58
 - coding alternatives, 59-61
 - Java clients, 63-64
 - lifetime management, 666

- MSMQ/Windows Communication Foundation integration, 407-411
- Open() method, 666
- scoping service hosts/clients, 666
- ClientViewOfData type, XmlFormatter example, 99**
- Close() method, 655-656, 664
- COM (Component Object Model), 21**
- COM+ (Component Object Model+), integration with, 385**
 - authentication/authorization requirements, 386
 - calling Windows Communication Foundation services from
 - building clients, 400-401
 - building services, 397-399
 - building VBScript files, 401-402
 - testing, 402
 - COM+ hosting mode, 387
 - COM+ Service Model Configuration tool, 387-389
 - component methods, 386
 - contacts, 386
 - supported interfaces, 386
 - usage example, 390-396
 - web hosting mode, 387
 - Windows Forms client references, 395-396
- Common Intermediate Language, 21**
- Common Language Infrastructure Specification, 21**
- Common Language Runtime, 21**
- communication state machines, 460**
- CommunicationObject property, 467**
- communications security, basic tasks of, 217**
- compensation, activities (Windows Workflow Foundation), 152-153**
- component methods, COM+ integration, 386**
- component-oriented programming, 21**
- composability (web services), defining, 413**
- composite activities (Windows Workflow Foundation), 141-142**
- CompositeDuplexBindingElement, 507**
- ComSvcConfig.exe (COM+ Service Model Configuration tool), 387-389**
- conditions, rules as (Windows Workflow Foundation), 181-183**
- confirmations, receiving messages, 116**
- Consistent Experience Across Contexts (Laws of Identity), 258**
- constructed types, 9**
- consuming services, 190-195**
- contacts, COM+ integration, 386**
- contracts, 26**
 - behavioral contracts, 652
 - callback contracts, 506-512
 - contract-first development, 83-84, 88
 - data contracts, 79-81, 650
 - defining
 - for DerivativesCalculator example, 34-40
 - XmlFormatter example, 96
 - explicit communication semantics, 458
 - fault contracts, 652
 - identifying/documenting service model items, 647
 - implementing, 27
 - implicit communication requirements, 458
 - interfaces, designating as, 27
 - matching custom channels to, 458-460
 - message contracts, 651
 - names, altering, 37
 - operation contracts, asynchronous pattern for, 662-664
 - Peer Channel, 544

- Principle of Scenario-Driven Design, 647-648
- public API layers, 647
- purpose of, 77
- REST POX service endpoints, 573
- scenarios, designing via, 646
- service contracts, 648-649, 653
- structural contracts, 649-651
- Correct member (Peer Channel example), 551**
- counters, performance, 659**
- custom activities, 136 (Windows Workflow Foundation)**
 - adding properties, 137
 - basic activities, 136-140
 - binding, 139-140
 - composite activities, 141
 - custom composite activities, 142
 - DependencyProperty property, 138-140
 - promoting properties, 140
 - XAML properties, 137
- custom behaviors**
 - attaching to operations/endpoints, 425-427, 449-450
 - declaring, 421-426
 - dispatchers, 420
 - informing Windows Communication Foundation of, 426-435, 450
 - instance context provider, service applications, 445-446
 - instance provider, service applications, 446-447
 - message formatter, 440-442
 - message inspector, 442-444
 - operation invokers, service applications, 447
 - operation selector, 436-437
 - parameter inspector, 438-440
 - usage examples, 451
 - WSDL export extension, 448-450
- custom binding elements**
 - applying through configuration, 476-480
 - inbound communication support, 470-476
 - outbound communication support, 463-470
 - understanding starting points, 461-462
- custom channels**
 - applying through configuration, 476-480
 - binding elements, 454-456
 - inbound communication support, 470-476
 - matching contracts to, 458-460
 - outbound communication support, 463-470
 - session support, 458
 - shapes of, declaring, 457
 - shapes of, 456-457
 - state machines, 460
 - understanding starting points, 461-462
- custom composite activities (Windows Workflow Foundation), 142**
- custom message inspectors, 646**
- custom operation selectors, 646**
- custom peer resolver services, 645**
- custom root activities (Windows Workflow Foundation), 165**
- custom services, workflow hosting, 177-179**
- custom streams, 526-535**
- custom transports, 481**
 - binding elements, 484-500
 - channel listeners, 493-496
 - message encoders, 499
 - TCP protocol support, 488-500
- CustomStream class, 527-531**

D

- data, large amounts of, 669**
- data contracts, 79-81**
 - adding optional members to, 627-628
 - defining, 99-102, 650
 - exception handling, 103-107
 - parameters, changing in, 626-630
- data representation**
 - background on, 77-79
 - exception handling, 103-107
 - XmlFormatter
 - advantages of, 82-83
 - contracts, defining, 96
 - data contracts, defining, 99-101
 - endpoints, adding programmatically, 95-96
 - overview, 79-81
 - Program.cs contents, 88
 - XmlSerializerFormat, 82
- DataContract, 79, 85**
- DataMember, 79, 85**
- DataPoint class, 507**
- dead-letter queues, 116-117**
- debugging applications, 671-672**
- decision trees, versioning, 624-625**
- DefaultLoaderService, 175**
- Delay activity (Windows Workflow Foundation), state machine workflows, 164**
- delegating maintenance, Windows Communication Foundation integration, 640-641**
- deleting operations, versioning, 630**
- DependencyProperty property, custom activities, 138-140**
- deploying services, DerivativesCalculator example, 49-52**
- DerivativesCalculation Class, 79-80**
- DerivativesCalculation data contract, 81**
- DerivativesCalculator example, 31, 34**
 - addresses, 42-48
 - binding, 42-48
 - clients
 - coding alternatives, 59-61
 - using with, 53-58
 - contracts, defining, 34-40
 - deploying the service, 49-52
 - hosting the service, 40-42, 65-67, 70
 - Java clients, 63-64
 - Program.cs file, 55-56
- DerivativesCalculator.csproj class library, 301-302**
- DerivativesCalculatorServiceType.cs, 315**
- DerivedData class, XmlFormatter example, 101-102**
- Details section (.ini files), Managed Cards, 330**
- digital identity (Identity Metasystem), 260-261**
- Directed Identity (Laws of Identity), 258**
- dispatch message formatters, 423**
- dispatchers, 78, 420**
- DispatchOperation property, 423**
- Distributed Transaction Coordinator (Microsoft), 13. See also Lightweight Transaction Manager**
- distributed transactions, 121**
- DivideByZero() method, exception handling, 104**
- DNS (domain name servers), CardSpace host file updates, 298**
- DSLs (domain-specific languages), 25-26**
- duplex communication, 645**
- duplex contracts, Peer Channel, 544**
- duplex service contracts, 653**
- DuplexChannel property, 457**

E

EnableSsl property, 669

encapsulation, 19

encoders (message), 482

- binding elements, implementing, 484-487
- custom transports, 499

ending/initiating sessions, 114

EndpointBehaviors property, 425

endpoints. See also Service Model [NPN]

- adding programmatically, XmlFormatter example, 95-96
- attaching custom behaviors to, 425-427, 449-450
- Peer Channel, 539, 543-544, 555-557
- REST POX services, 572-573
- retiring, versioning, 631
- service endpoints, changing addresses of, 631

error handling, activities (Windows Workflow Foundation), 134

evaluation context (claim sets), 339, 362

EventDrivenActivity activity (Windows Workflow Foundation), state machine workflows, 164

exception handling, data contracts, 103-107

exceptions

- anticipating, 664-665
- CardSpace (Windows), catching in, 313-314

exchange patterns, designing, 551

Execute() method, custom activities, 136

exemplars, Windows Communication Foundation integration, 639

explicit communication semantics, 458

export extension (WSDL)

- attaching to operations/endpoints, 449-450
- declaring, 448
- informing Windows Communication Foundation of, 450

Extension property, 656-658

F - G

Fabrikam certificates, 300, 309

fault contracts, 652

Faulty operation, exception handling, 105-106

federations (security), 342

- claim-based security versus, 380
- security token services, 288

FFIEC (Federal Financial Institutions Examination Council), authentication, 255

Fielding, Dr. Roy, web browser architectures, 570

formatname: MULTICAST, 518

Formatter property, 422-423

Forms client, COM+ integrations, 395-396

forward chaining, 186

frameworks

- API layers, designing public layers, 647
- layered designs, 419
- Principle of Scenario-Driven Design, 647-648

generics, 8

- base types, 10
- constructed types, 9
- methods, 9-10

- type arguments, 9-10

- type definitions, 9-12

- type parameters, 9

GetCallbackChannel<T>() generic method, callback contracts, 507

GetKnownDataPoints() operation, callback contracts, 508

GetPicture() operation

- custom streams, 527-531

- streamed transfer mode, 522, 526

GetProperty<T> property, 464

GetToken() method, CardSpace, 275

H

HandleExternalEvent activity

- state machine workflows, 164

- workflow communication, 143, 147-149

health models, services, 659-660

high assurance, 307

high value assurance certificates, 307

Host.csproj console application, 303-305

hosting

- services, 28

- workflows, 166-167

- inside WCF services, 200-205

- runtime services, 168-179

hosting service

- DerivativesCalculator example, 40-42

- IIS, 65-67, 70

HTTP (Hypertext Transfer Protocol)

- binding elements and, 46

- CardSpace (Windows), configuring in, 300-301

HTTP bindings, 644, 670

Human Integration (Laws of Identity), 258

I

identity

- FFIEC (Federal Financial Institutions Examination Council), 255

- Identity Metasystem, 259

- anonymous identity, 260

- architecture of, 262

- claims, 260

- digital identity, 260-261

- Identity Providers, 262-263, 266

- Identity Selector, 265-266

- Information Cards, 266-273

- Liberty identity protocols, 264

- Relying Parties, 262, 265

- requirements, 263

- roles of, 261

- security tokens, 261

- Triangle of Trust, 287

- WS-Security, 264

- WS-Trust, 264

- Laws of Identity, 257-259

- Passport (Microsoft), 257

- passwords, 255

- PKI (Public Key Infrastructure), 257

- regulatory compliance, 256

- role of, 253-255

- SSO (single sign on), 256, 286

- user-centric identity management, 259

Identity (Windows), authorizing resource access, 343-350

Identity Metasystem, 259

- anonymous, 260

- architecture of, 262

- CardSpace (Windows), 266-267, 286-289
 - adding Information Cards, 268-270, 301-313, 317-328
 - architecture of, 273-276
 - authentication methods, 280-285
 - card metadata, 273
 - catching exceptions, 313-314
 - Choose a Card dialog, 279
 - editing cards, 272
 - enabling ASP.NET 2.0, 296-297
 - GetToken() method, 275
 - HTTP configuration, 300-301
 - IIS (Internet Information Services), 296-298
 - Introduction dialog, 276-277
 - IssuedToken policy assertion, 275
 - Metadata Resolver, 316-317
 - mutual authentication dialog, 307
 - passwords, 280
 - preview dialog, 268
 - processing security tokens, 314-316
 - protocols, 278
 - security tokens, 273, 281
 - usage example, 295
 - X.509 certificates, 297-299
- claims, 260
- digital identity, 260-261
- Identity Providers, 262-263
 - building, 293
 - Information Cards, 266
- Identity Selector, 265-267
- Information Cards, 266-267
 - adding/installing, 268-270
 - editing, 272
 - Managed Cards, 269
 - metadata, 273
 - Personal Cards, 268-270
 - revoking, 279
 - security tokens, 273
 - Self-Issued Cards, 268, 272
- Liberty identity protocols, 264
- Relying Parties, 262, 265, 294
- requirements, 263
- roles of, 261
- security tokens, 261
- Triangle of Trust, 287
- Windows clients, 295
- WS-Security, 264
- WS-Trust, 264
- Identity Providers, 262-263, 266, 293**
- Identity Selector**
 - CardSpace (Windows), 266-267, 286-289
 - adding/installing cards, 268-270
 - architecture of, 273-276
 - authentication methods, 280-285
 - card metadata, 273
 - Choose a Card dialog, 279
 - editing cards, 272
 - GetToken() method, 275
 - Introduction dialog, 276-277
 - IssuedToken policy, 275
 - passwords, 280
 - preview dialog, 268
 - protocols, 278
 - security tokens, 273, 281
 - Identity Metasystem, 265-267
- IfElse activity (Windows Workflow Foundation), sequential workflows, 155**
- IIS (Internet Information Services)**
 - CardSpace (Windows), 296-298
 - hosting services, 65-67, 70
 - service hosting, 655

impersonation, 236-239

implicit communication requirements, 458

importing X.509 certificates to CardSpace certificate stores, 297-298

InfoCard. See CardSpace (Windows)

infocard.exe (CardSpace), 273

infocardapi.dll (CardSpace), 274

Information Cards, 266-267

adding to, 268-270

browser applications, 317-328

WCF applications, 301-316

building STS (Security Token Services), 332-334

editing, 272

Managed Cards, 269, 328-332

Metadata Resolver, 316-317

metadata, 273

Personal Cards, 268-270

Provider Cards, 269

revoking, 279

security tokens, 273

Self-Issued Cards, 268, 272

informing Windows Communication

Foundation of custom behaviors, 426-435, 450

InfoSet (XML Information Set), data representation, 78

.ini files (Managed Cards), 330-331

initiating/ending sessions, 114

InnerProxy property, ClientBase<T>, 60

InputChannel property, 456-457

instance context provider custom behavior, service applications, 445-446

instance context sessions, 113

instance provider custom behavior, service applications, 446-447

InstanceContextProvider property, 423

InstanceProvider property, 423

integration

COM+, 385

authentication/authorization requirements, 386

calling Windows Communication Foundation services from, 397-402

COM+ hosting mode, 387

COM+ Service Model Configuration tool, 387-389

component methods, 386

contacts, 386

supported interfaces, 386

usage example, 390-396

web hosting mode, 387

Windows Forms client references, 395-396

MSMQ

creating clients, 407-411

creating requests, 403-404

creating services, 404-407

testing applications, 411

Windows Communication Foundation

delegating maintenance, 640-641

exemplars, 639

incorporating existing applications, 640

rebuilding existing applications, 639

interfaces

contracts, designating interfaces as, 27

defining with C++, 20-21

writing in .NET, 27

interoperability, 413-415

Introduction dialog (CardSpace), 276-277

InvokeWebService activity (Windows Workflow Foundation), 190-191

IOrderSubscriber interface, 515

IOrderSubscriber service contract (MSMQ PGM), 520

IPictureServer

- custom streams, 527-531
- streamed transfer mode, 522-523, 526

IPublisher, callback contracts, 506-512

IPv6 NAT Traversal, Teredo, 538

ISecurityTokenService service contracts, STS addition process, 360

IssuedToken policy assertions (CardSpace), 275

Issuer section (.ini files), Managed Cards, 330

ISubscriber

- callback contracts, 506-512
- stream transfer mode, 532

Item member (Peer Channel example), 551

IUnknownSerializationData interface, XmlFormatter example, 99

J - K - L

Java clients, DerivativesCalculator example, 63-64

Java Virtual Machine Specification, class file format, 21

Justifiable Parties (Laws of Identity), 258

Kerberos authentication method, 280

KnownType attribute, XmlFormatter example, 102-103

Laws of Identity, 257-259

layered framework designs, 419

legacy integration

COM+, 385

- authentication/authorization requirements, 386
- calling Windows Communication Foundation services from, 397-402
- component methods, 386
- contacts, 386
- hosting mode, 387
- Service Model Configuration tool, 387-389
- supported interfaces, 386
- usage example, 390-396
- web hosting mode, 387
- Windows Forms client references, 395-396

MSMQ, 403

- creating clients, 407-411
- creating requests, 403-407
- testing applications, 411

Liberty identity protocols, 264

Lightweight Transaction Manager, 13-14. See also Distributed Transaction Coordinator (Microsoft)

Limited Disclosure for a Constrained Use (Laws of Identity), 258

logotypes, adding Information Cards to WCF applications, 308

M

Main() method (MSMQ PGM), 516-520

maintenance, delegating (Windows Communication Foundation integration), 640-641

Managed Cards, 269, 328-332

Management Model designer, 621

ManualWorkflowSchedulerService, 174-175

maxMessageSize property, streamed transfer mode, 525

Message class, data representation, 78

message encoders, 482

- binding elements, 484-487
- custom transports, 499

message formatter custom behavior, 440-442

message inspector custom behavior, 442-444, 646

Message object, data representation, 78

Message Transmission Optimization Mechanism. See MTOM

MessageEncoding property, 670

MessageFormatter property, 425

MessageInspectors property, 422-423

messages, 20

- contracts, 651
- defining with C++, 20-21
- designing, Peer Channel example, 551
- exchange patterns, 457
- logging, 71, 659-661
- routing, Windows Workflow Foundation workflows inside WCF services, 200-202
- security, 218, 230
 - client configuration, 231-232
 - service endpoint configuration, 233-236

Metadata Resolver, 61, 316-317

methods

- generic, 9-10
- one-way methods, transaction execution, 119

Microsoft Management Model designer, 621

Microsoft Message Queuing. See MSMQ (Microsoft Message Queuing)

Microsoft Password, 257

Microsoft Windows Vista DEBUG Build Environment prompt, 53

model-driven software development, 24-26

Mono, 21

MSMQ (Microsoft Message Queuing), 114

- base address schemes, 48
- integration with
 - creating clients, 407-411
 - creating requests, 403-404
 - creating services, 404-407
 - testing applications, 411
- queued delivery, 115-118

MSMQ PGM, 513-516, 520-521

MsmqIntegrationBinding, 47, 518, 521

MsmqMessage<PurchaseOrder>, 516-517

MsmqMessage<T> types, 521

MTOM (Message Transmission Optimization Mechanism), 46, 670

mutual authentication dialog (CardSpace), 307

MyChannelListener.cs, 471

MyCustomBindingElement.cs, 463-464, 469-470, 476

MyCustomReplyChannel.cs, 472

MyCustomRequestChannel.cs, 466

N

Name parameter, 37

Named Pipes, base address schemes, 48

names (contract), altering, 37

Namespace parameter, 37

NAT-T, Teredo, 538

.NET Framework Class Library, service-oriented programming, 24

NetMsmgBinding, 643-644

MSMQ PGM, 521

queued delivery, 115

session management, 113

NetMsmqBinding, 47

NetNamedPipeBinding, 47, 584, 644

Reliable Sessions, 110

session management, 113

transaction execution, 120

NetPeerTcpBinding, 47, 645

NetTcpBinding, 47, 485, 584, 644-645

CompositeDuplexBindingElement, 507

Reliable Sessions, 110

session management, 113

transaction execution, 120

networks, P2P (peer-to-peer), 539

NetworkStream class, custom transports, 486

NextValueHandler() method, callback contracts, 510

non-backwards-compatible changes (versioning), 624

normalization (claims), 342

NotificationStreamWriter class, 532-535

Notify() operation

callback contracts, 506, 511

MSMQ PGM, 517, 520

streamed transfer mode, 534

nullable value types, 11-12

O

OleTx protocol, 118

one-way methods, transaction execution, 119

one-way operations (callback contracts), 506

Open() method, 520, 656, 666

operation contracts, asynchronous pattern for, 662-664

operation invokers custom behavior, service applications, 447

operation selectors, custom behavior, 436-437, 646

OperationBehaviors property, 425

OperationContext class, callback contracts, 510

OperationContract attributes

Action/ReplyAction parameters, 360

altering contract names, 37

MSMQ PGM, 516

OperationInvoker property, 423

operations, attaching custom behaviors to, 425-427, 449-450

OperationSelector property, 422-423

orchestrating services, 195-196

Order project (MSMQ PGM), 514

OrderPullPoint configuration (MSMQ PGM), 517-518

OrderSubscriber class, MSMQ PGM, 519-520

out of the box activities (Windows Workflow Foundation), 135-136

OutputChannel property, 456-457

P

P2P (peer-to-peer) networks, 539

Parallel activities, 134, 157-163

parameter inspector custom behavior, 438-440

ParameterInspectors property, 422-423

parameters, changing data contracts, 626-630

partial types, usage examples, 7

Password (Microsoft), 257

passwords, 255, 280

Peer Channel

binding, 540-543

contracts, 544

custom peer name resolution, 560-565

directing messages to specific peers,
558-560

endpoints, 539, 543-544, 555-557

example

communication, implementing, 554

designing messages/message
exchange patterns, 551

overview, 545-549

testing, 566-567

People Near Me, 568

reasons for, 537-538

service contracts, 552-554

Windows Peer-to-Peer Networking fea-
tures, 538

peer-to-peer applications, structured data in,
537-538. *See also* Peer Channel

People Near Me, 568

performance counters, 659

persistence services, workflow hosting,
168-170

Personal Cards (Information Cards), 268-270

PGM. *See* MSMQ PGM

phishing attacks, 254

PictureBox control, 524

PictureServer class, streamed transfer mode,
522-523

PKI (Public Key Infrastructure), 257

PlaceOrder activities, 134

Pluralism of Operators and Technologies
(Laws of Identity), 258

PNRP (Peer Name Resolution Protocol), 538

poison queues, 117-118

policies

assertions, 632

centralized lifecycle management, 632

rules as (Windows Workflow Foundation),
185-187

PowerShell scripts, 659

predefined bindings, 643-645

Principle of Scenario-Driven Design, 647-648

ProcessMessage() method, 405

Program.cs file

contents of, 88

DerivativesCalculator example, 55-56

promoting

properties in activities (Windows
Workflow Foundation), 140

transactions, 13

proof keys, 281

properties

adding to custom activities, 137

DependencyProperty, custom activities,
138-140

promoting in activities (Windows Workflow
Foundation), 140

XAML properties in custom activities,
137

Provider Cards (Information Cards), 269

public API layers (frameworks), designing,
647

public key infrastructure. *See* PKI (Public Key
Infrastructure)

publish/subscribe, 505-506, 531-535

Publisher class (MSMQ PGM), 516-517

Publisher project (MSMQ PGM), 514

PublisherService class, 507, 532-534

PublisherService service type, callback con-
tracts, 510

publishing web services, 196-199

PublishingAgent class, streamed transfer mode, 532-534

pull-style notification, 505, 514

PurchaseOrder class (MSMQ PGM), 514

pure virtual functions, abstract base classes, 21

push-style notification, 505

Q - R

queued delivery, 114

- dead-letter queues, 116-117
- NetMsmqBinding, 115
- poison queues, 117-118
- private messages, 115
- receiving messages, 116
- usage example, 122-128

queueing service requests, 643

QuizItem struct (Peer Channel example), 551

QuizResponse struct (Peer Channel example), 551

RandomDataPoint class, 507

RandomDataPoint project, 507

Read() method, custom streams, 529-534

Really Simple Syndication. *See* RSS

Reliable Sessions, 109

- custom bindings, 110
- supported bindings, 110
- toggling on/off, 110
- usage example, 111-113

Relying Parties

- building, 294
- Identity Metasystem, 262, 265

ReplyAction parameter (**OperationContract** attributes), 360

ReplyChannel property, 457-459, 470-474

RequestChannel property, 457-459, 465-468

requests, MSMQ/Windows Communication Foundation integration, 403-404

resolver services, 645

Resource Access client

- resource access, authorizing, 343-350
- STS addition process
 - certificate installation, 358-359
 - Fabrikam STS authorization policies, 360-363
 - ISecurityTokenService service contracts, 360
 - prebuilt STS, 359
 - Resource Access Service reconfiguration, 372-376
 - Woodgrove STS, 364-372

Responder member (Peer Channel example), 551

ResponseText member (Peer Channel example), 551

REST (Representational State Transfer), REST POX services, 569

- building RSS servers, 574-579
- endpoint addresses, 572
- endpoint bindings, 572-573
- endpoint contracts, 573
- implementing, 574
- SOAP services versus, 571
- XML, 570

retiring endpoints, versioning, 631

RetrievePicture() method, streamed transfer mode, 523

ReturnExceptionDetailInFaults, 671-672

revoking Information Cards, 279

rights (claims), 338

role providers, 14

- configuring, 16, 240-241

- System.Web.Security.RoleProvider, 15

roles (security)

- definition of, 339

- role-based authorization

- claim-based authorization versus, 339-340

- resource access, authorizing, 343-350

Root Agency, transport security, 219-221**routing messages, Windows Workflow**

- Foundation workflows inside WCF services, 200-202

RSS (Really Simple Syndication) servers, building, 574-579**rules engine, Windows Workflow Foundation, 179**

- CAG (ConditionedActivityGroup) activity, 184

- rules

- as conditions, 181-183

- as policies, 185-187

runtime services, workflow hosting

- custom services, 177-179

- persistence services, 168-170

- scheduler services, 174-177

- tracking services, 170-173

- workflow hosting, 168

S**SAML (Security Assertion Markup Language), claim-based authorization, 341****scheduler services, workflow hosting**

- DefaultLoaderService, 175

- ManualWorkflowSchedulerService, 174-175

- SharedConnectionWorkflowCommitWorkBatchService, 176-177

- WorkflowQueuingService, 176

scoping service hosts/clients, 666**security**

- authorization, 239, 242, 248

- configuring services to use role providers, 240-241

- service configuration, 243-245

- ServiceAuthorizationManager implementation, 246-247

- basic tasks of, 217

- CardSpace (Windows), 278, 286-289

- authentication methods, 280-285

- Choose a Card dialog, 279

- passwords, 280

- security tokens, 281

- claim sets, 338

- claims

- claim-based authorization, 339-341, 378-379

- description of, 338

- federated security versus, 380

- normalization, 342

- rights of, 338

- role-based authorization, 350-351, 354-357

- SAML, 341

- types of, 338

- values of, 338

- events, auditing, 661

- federations, 342, 380

- identity

- FFIEC (Federal Financial Institutions Examination Council), 255

- Identity Metasystem, 259-266, 287

- Laws of Identity, 257-259

- regulatory compliance, 256
- role of, 253-255
- user-centric identity management, 259
- Identity Metasystem, 266-273
- impersonation, 236-239
- message, 218, 230
 - client configuration, 231-232
 - service endpoint configuration, 233-236
- overview, 217
- Password (Microsoft), 257
- passwords, 255
- phishing attacks, 254
- PKI (Public Key Infrastructure), 257
- roles
 - definition of, 339
 - role-based authorization, 343-350
- SID (Security Identifiers), 340
- SSO (single sign on), 256, 286
- STS addition process, 341
 - certificate installation, 358-359
 - client reconfiguration, 376-378
 - Fabrikam STS configurations, 360-363
 - ISecurityTokenService service contracts, 360
 - prebuilt STS, 359
 - Resource Access Service reconfiguration, 372-376
 - Woodgrove STS, 364-372
- tokens
 - adding Information Cards to WCF applications, 307
 - CardSpace, 281
 - federation, 288
 - Identity Metasystem, 261
 - processing in CardSpace (Windows), 314-316
- transport, 218
 - configuring server identities, 223
 - determining certificate trustworthiness, 219-221
 - identifying server provided certificates, 221-222
 - installing certificates, 219-221
 - removing SSL configurations, 250
 - restoring server identities, 250-251
 - specifying certificates in SSL exchanges, 221-222
 - SSL protocol, 219
 - uninstalling certificates, 249
 - usage example, 223-229
- Security Assertion Markup Language. See SAML**
- security identifiers. See SID**
- security token service. See STS**
- Self-Issued Cards (Information Cards), 268, 272**
- SendResponse message, defining service contracts, 552**
- sequential workflows, Windows Workflow Foundation, 155-163**
- Serializable attribute (PurchaseOrder class), 514**
- Serialization, Program.cs contents, 88**
- Service directive, IIS hosting, 66**
- Service Metadata Tool. See SvcUtil.exe**
- Service model, 25**
 - addresses, 26-28, 42-48
 - binding, 26-28, 42-48
 - contracts, 26
 - defining, 34-40
 - designating interfaces as, 27
 - implementing, 27
 - deploying services, 49-52

- DerivativesCalculator example, 31-34
 - clients, coding alternatives, 59-61
 - clients, using with, 53-58
 - Java clients, 63-64
- hosting services, 28
 - DerivativesCalculator example, 40-42
 - in IIS, 65-67, 70
- identifying/documenting items, 647
- overview, 26-27, 31, 77
- service operators, including in transactions, 120**
- service types, 27-28, 38-39**
- service-oriented programming, 22-24**
- ServiceBehavior attribute, 27**
- ServiceContract attribute, 516**
 - callback contracts, 506
 - contract names, altering, 37
 - service types and, 38
- ServiceHost class, 28, 520**
- ServiceHost instance, MSMQ PGM, 520**
- services. See also Service Model [NPN]**
 - COM+/Windows Communication Foundation integration, 397-399
 - configuration, 124
 - contracts
 - defining, 552-554
 - designing, 648-649
 - duplex contracts, 653
 - implementing, Peer Channel, 554
 - inheritance, 625-626
 - definition of, 77
 - exposing workflows as, 196-199
 - hosting Windows Workflow Foundation workflows as
 - message routing, 200-202
 - runtime hosting options, 203-205
 - MSMQ/Windows Communication Foundation integration, 404-407
 - processing, 124-126
 - requests, queueing, 643
 - REST POX services, 570
 - building RSS servers, 574-579
 - endpoint addresses, 572
 - endpoint bindings, 572-573
 - endpoint contracts, 573
 - implementation, 574
 - SOAP services versus, 571
 - System.Security.Permissions.PrincipalPermission, 658
 - System.ServiceModel.ServiceAuthorizationManager, 658
 - Windows Communication Foundation
 - activity tracing, 659-661
 - ASPNET Compatibility Mode, 658
 - auditing security events, 661
 - authorization, 658
 - class libraries, 656
 - comprehensive health models, 659-660
 - ensuring manageability, 658
 - Extension property, 656-658
 - IIS hosting, 655
 - message logging, 659-661
 - performance counters, 659
 - PowerShell scripts, 659
 - session state information, 656
 - synchronizing shared data access, 654
 - WMI provider, 659
 - XML documents, describing in, 26
- ServiceViewOfData class, XmlFormatter example, 98-99**

session management

- initiating/ending sessions, 114
- instance context sessions, 113
- supported bindings, 113
- usage example, 122-128

Session parameter (ServiceContract attribute), callback contracts, 506**SetState activity (Windows Workflow Foundation), state machine workflows, 164****shared data, synchronizing access, 654****SharedConnectionWorkflowCommitWorkBatchService, 176-177****SID (security identifiers), 340****Singleton services, Windows Workflow Foundation workflows inside WCF services, 203-205****SOAP (Simple Object Access Protocol)**

- binding elements and, 46
- messages, streamed transfer mode, 526
- MTOM (Message Transmission Optimization Mechanism), 670
- REST POX services versus, 571

SOAP Toolkit, service-oriented programming, 24**software factory templates, 24-26. See also model-drive software development****SqlTrackingService, workflow hosting, 173****SSL (Secure Sockets Layer) protocol, transport security, 219****SSO (single sign on), 256, 286****standard bindings, 46-48****state machine workflows, 163-165, 460****streamed transfer mode, 670-671**

- custom streams, 526-531
- overview, 521-523, 526
- publish/subscribe, 531-535

StreamingPublicationSubscription library, 532**structs (Peer Channel example), 549-551****structural contracts, 649-651****STS (security token services), 341. See also XSI [NPN]****addition process**

- certification installation, 358-359
- client reconfiguration, 376-378
- Fabrikam STS configuration, 360-363
- ISecurityTokenService service contracts, 360
- prebuilt STS, 359
- Resource Access Service reconfiguration, 372-376
- Woodgrove STS, 364-372

building, 332-334**Student struct (Peer Channel example), 551****subject confirmation keys, 281****Submitted member (Peer Channel example), 551****Subscribe() method**

- callback contracts, 508
- streamed transfer mode, 532-534

subscriber configuration (MSMQ PGM), 518-519**.svc, 66****SvcUtil.exe, DerivativesCalculator example, 53-58****synchronizing shared data access, 654****System.InvalidOperationException exceptions, matching contracts to custom channels, 458****System.Net.FtpWebRequest, 669****System.Nullable<T>, 11-12****System.Runtime.Serialization assembly, data representation. See XmlFormatter****System.Security.Permissions.PrincipalPermission, 240-243, 658****System.ServiceModel.Channel.HttpTransportBindingElement, 477**

- System.ServiceModel.Channels, 420
- System.ServiceModel.Channels.Binding-Context, 459
- System.ServiceModel.Channels.Binding-Element, 463
- System.ServiceModel.Channels.Channel-ListenerBase<IReplyChannel>, 471
- System.ServiceModel.Channels.IChannel interface, 456
- System.ServiceModel.Channels.ICommunicationObject interface, 460, 467
- System.ServiceModel.Channels.ICommunicationObject, 474
- System.ServiceModel.Channels.IDuplex-Channel, 457
- System.ServiceModel.Channels.IInputChannel, 456-457
- System.ServiceModel.Channels.IOutput-Channel, 456-457
- System.ServiceModel.Channels.IReply-Channel interface, 470-471
- System.ServiceModel.Channels.IReply-Channel, 457-459
- System.ServiceModel.Channels.IRequest-Channel interface, 465-468
- System.ServiceModel.Channels.IRequest-Channel, 457-459
- System.ServiceModel.Channels.ISession interface, 458
- System.ServiceModel.Channels.Reliable-SessionBindingElement, 110
- System.ServiceModel.Communication-Exception, 664
- System.ServiceModel.Configuration.Binding-ElementExtensionElement, 478-479
- System.ServiceModel.Description.IEndpoint-Behavior, 425
- System.ServiceModel.Description.IOperation-Behavior, 425
- System.ServiceModel.Dispatcher, 420-421
- System.ServiceModel.Dispatcher.Channel-Dispatcher, 422
- System.ServiceModel.Dispatcher.Client-Operation, 421
- System.ServiceModel.Dispatcher.Client-Runtime, 421-422
- System.ServiceModel.Dispatcher.Dispatch-Operation, 422-423
- System.ServiceModel.Dispatcher.Dispatch-Runtime, 422
- System.ServiceModel.Dispatcher.IClient-MessageFormatter interface, 421
- System.ServiceModel.Dispatcher.IClient-MessageInspector, 422
- System.ServiceModel.Dispatcher.IDispatch-MessageFormatter, 423
- System.ServiceModel.Dispatcher.IDispatch-MessageInspector, 423
- System.ServiceModel.Dispatcher.IDispatch-OperationSelector, 423
- System.ServiceModel.Dispatcher.IInstance-ContextProvider, 423
- System.ServiceModel.Dispatcher.IInstance-Provider, 421-423
- System.ServiceModel.Dispatcher.IOperation-Invoker, 423
- System.ServiceModel.Dispatcher.IParameter-Inspector, 422-423
- System.ServiceModel.IReplyChannel interface, 473-474
- System.ServiceModel.OperationContext object, 119
- System.ServiceModel.ServiceAuthorization-Manager, 658
- System.ServiceModel.ServiceDebugBehavior, ReturnExceptionDetailInFaults, 671-672
- System.ServiceModel.ServiceHost, Close() method, 655-656
- System.ServiceModelDispatcher.IClient-MessageFormatter, 425
- System.TimeoutException, 664

System.Transactions namespace, including in service operator transactions, 120

System.Web.Security.RoleProvider, 15

System.Xml assembly, data representation.

See **XmlSerializerFormat**

T

TCP (Transfer Control Protocol)

base address schemes, 48

binding elements and, 46

TcpClient class, custom transports, 485-487

TcpListener class, custom transports, 485-487

Teredo, 538

testing

COM+/Windows Communication Foundation integrations, 402

MSMQ/Windows Communication Foundation integration applications, 411

TokenDetails section (.ini files), Managed Cards, 331

TokenTypes section (.ini files), Managed Cards, 331

ToString() method (MSMQ PGM), 514

tracking services, workflow hosting, 170-173

transactions

committing, 119-120

distributed transactions, 121

OleTx protocol, 118

one-way methods, 119

promotion of, 13

service operators, including in, 120

supported bindings, 120

usage example, 122-128

WS-AT (WS-AtomicTransaction) protocol, 118

transports

customizing, 481

binding elements, 500

channel listeners, 493-496

implementing binding elements, 484-500

message encoders, 499

TCP protocol support, 488-500

inbound communication, 482-483

message encoders, 484-487

message encoders, 482

outbound communication, 482-483

security, 218

certificates, 219-222, 249-250

server identities, 223, 250-251

SSL protocol, 219

usage example, 223-229

Triangle of Trust (Identity Metasystem), 287

Trusted Root Certification Authority Store, installing transport security certificates, 219-221

types (claims), 338

U - V

UML (Unified Modeling Language), 24

User Control and Consent (Laws of Identity), 258

user-centric identity management, 259

validating activities, 150-151

value types (nullable), 11-12

values (claims), 338

VBScript files, COM+/Windows Communication Foundation integration, 401-402

versioning

- adding operations, 625-626
- backwards-compatible changes, 624
- centralized lifecycle management, 632-634
- changing
 - bindings, 630
 - operations, changing parameter data contracts, 626-630
 - service endpoint addresses, 631
- decision tree, 624-625
- deleting operations, 630
- non-backwards-compatible changes, 624
- retiring endpoints, 631
- service contract inheritance, 625-626

Vista

- dead-letter queues, 116-117
- poison queues, 117-118

Visual Studio .NET, Windows Forms Designer, 25**W****web browsers**

- adding Information Cards to, 317-328
- architectures of, 570

web services

- composability, 413
- defining, 413
- interoperability, 413-415
- publishing, 196-199

Web Services Description Language. See WSDL**Web Services Eventing, publish/subscribe systems and, 505****Web Services Notification, 505****Web Services Trust Language. See WS-Trust****Windows clients (Identity Metasystem), building, 295****Windows Communication Foundation**

- addresses, 641-642
- bindings, 643-646
- clients
 - AJAX, 667-668
 - anticipating exceptions, 664-665
 - asynchronous pattern for operation contracts, 662-664
 - lifetime management, 666
 - Open() method, 666
 - scoping service hosts/clients, 666
- contracts
 - behavioral contracts, 652
 - defining data contracts, 650
 - designing public API layers, 647
 - designing service contracts, 648-649
 - designing via scenarios, 646
 - duplex service contracts, 653
 - fault contracts, 652
 - identifying/documenting service model items, 647
 - message contracts, 651
 - Principle of Scenario-Driven Design, 647-648
 - service contracts, 653
 - structural contracts, 649-651
- debugging applications, 671-672
- integrating
 - delegating maintenance, 640-641
 - exemplars, 639
 - incorporating existing applications, 640
 - rebuilding existing applications, 639
- large amounts of data, 669

- MTOM, 670
- services
 - activity tracing, 659-661
 - ASP.NET Compatibility Mode, 658
 - auditing security events, 661
 - authorization, 658
 - class libraries, 656
 - comprehensive health models, 659-660
 - ensuring manageability, 658
 - Extension property, 656-658
 - IIS hosting, 655
 - message logging, 659-661
 - performance counters, 659
 - PowerShell scripts, 659
 - session state information, 656
 - synchronizing shared data access, 654
 - System.Security.Permissions.Principal Permission, 658
 - System.ServiceModel.ServiceAuthorizationManager, 658
 - WMI provider, 659
 - streaming transfer mode, 670-671
- Windows Forms client, COM+ integrations, 395-396**
- Windows Forms Designer, 25**
- Windows Server 2003, Authorization Manager, 344-350**
- Windows Workflow Foundation**
 - activities
 - ACID transactions, 152
 - binding, 139-140
 - CAG (ConditionedActivityGroup), 184
 - compensation, 152-153
 - custom activities, 136-142
 - custom root, 165
 - Delay, 164
 - design behavior, 150-151
 - error handling, 134
 - EventDrivenActivity, 164
 - HandleExternalEvent, 164
 - IfExists, 155
 - initializing, 134
 - InvokeWebService, 190-191
 - lifecycle of, 134
 - out of the box activities, 135-136
 - Parallel activities, 134
 - Parallel, 157-163
 - PlaceOrder activities, 134
 - promoting properties, 140
 - SetState, 164
 - validation, 150-151
 - workflow communication host configuration, 145-149
 - workflow communication interface, 143-145
 - Bind dialog, 140
 - components of, 132
 - consuming services, 190-195
 - defining, 132
 - orchestrating services, 195-196
 - rules engine, 179
 - CAG (ConditionedActivityGroup) activity, 184
 - rules as conditions, 181-183
 - rules as policies, 185-187
 - workflows, 154
 - exposing as services, 196-199
 - hosting, 166-179, 200-205
 - sequential workflows, 155-163
 - state machine workflows, 163-165
- WMI provider, 659**

WorkflowQueuingService, 176**workflows, 154**

activities

communicating via, 143-145

configuring host communication,
145-149

custom root activities, 165

exposing as services, 196-199

hosting, 166-167

inside WCF services, 200-205

runtime services, 168-179

sequential workflows, 155-163

state machine workflows, 163-165

WS-AT (WS-AtomicTransaction) protocol, 118**WS-FederationBinding, 643****WS-ReliableMessaging protocol, binding elements and, 46****WS-Security protocol, 46, 264****WS-Trust (Web Services Trust Language), 264, 341, 360****WSDL export extension, 22**attaching to operations/endpoints,
449-450

declaring, 448

downloading, 29

informing Windows Communication
Foundation of, 450

key terms, 26

WSDL2Java, 63-64**WSDualHttpBinding, 47, 645-646**

CompositeDuplexBindingElement, 507

Reliable Sessions, 110

session management, 113

transaction execution, 120

wsFederationBinding, 47, 414, 584adding Information Cards to WCF applica-
tions, 311-313

Reliable Sessions, 110

transaction execution, 120

WSHttpBinding, 47, 414, 584, 643

Reliable Sessions, 110

session management, 113

transaction execution, 120

X - Y - Z**X.509 certificates, CardSpace (Windows)**

host file updates, 298

importing into certificate store, 297-298

private key access, 299

XAML properties in custom activities, 137**XML (Extensible Markup Language)**

contract-first development and, 84, 88

REST services, 570

WSDL and, 22

XML Information Set (InfoSet), 78**XmlFormatter**

advantages of, 82-83

contract-first development and, 84, 88

example

contracts, defining, 96

data contracts, defining, 99-102

endpoints, adding programmatically,
95-96

Program.cs contents, 88

overview, 79-81

XmlSerializer, contract-first development and, 87-88

XmlSerializerFormat, 82**XSI**

Authorization Manager, 350-351,
354-357

claim-based authorization, 378-379

ACL versus, 340

adoption strategies, 341

authorizing resource access, 350-351,
354-357

role-based authorization versus,
339-340

STS addition process

certificate installation, 358-359

client reconfiguration, 376-378

Fabrikam STS authorization policies,
362-363

Fabrikam STS configurations, 360-362

ISecurityTokenService service con-
tracts, 360

prebuilt STS, 359

Resource Access Service reconfigura-
tion, 372-376

Woodgrove STS, 364-372