"Web services is the integration technology preferred by organizations implementing service-oriented architectures. I would recommend that anybody involved in application development obtain a working knowledge of these technologies, and I'm pleased to recommend Erl's book as a great place to begin."

—Tom Glover, Senior Program Manager, Web Services Standards, IBM Software Group, and Chairman of the Web Services Interoperability Organization (WS-I)

Service-Oriented Architecture

A Field Guide to Integrating XML and Web Services

Thomas Erl

PRENTICE HALL PTR

Contents

Preface XIX

Chapter 1

Introduction 1

1.1 Why this guide is important 2

1.1.1 The hammer and XML 2

1.1.2 XML and Web services 3

1.1.3 Web services and Service-Oriented Architecture 3

1.1.4 Service-Oriented Architecture and the hammer 3

1.1.5 The hammer and you 4

1.2 The XML & Web Services Integration Framework (XWIF) 4

1.3 How this guide is organized 5

1.3.1 Part I: The technical landscape 6

1.3.2 Part II: Integrating technology 7

1.3.3 Part III: Integrating applications 9

1.3.4 Part IV: Integrating the enterprise 12

1.3.5 The extended enterprise 13

1.4 www.serviceoriented.ws 13

1.5 Contact the author 13

Part I

The technical landscape

15

Chapter 2

Introduction to XML technologies

17

2.1 Extensible Markup Language (XML) 18
2.1.1 Concepts 20
2.1.2 Schemas 21
2.1.3 Programming models 22
2.1.4 Syntax 23
2.2 Document Type Definitions (DTD) 24
2.2.1 Concepts 25
2.2.2 Syntax 25
2.3 XML Schema Definition Language (XSD) 28
2.3.1 Concepts 28
2.3.2 Syntax 28
2.4 Extensible Stylesheet Language Transformations (XSLT) 33
2.4.1 Concepts 34
2.4.2 Syntax 35
2.5 XML Query Language (XQuery) 38
2.5.1 Concepts 38
2.5.2 Syntax 41
2.6 XML Path Language (XPath) 43
2.6.1 Concepts 43
2.6.2 Syntax 44

Chapter 3

Introduction to Web services technologies

47

3.1 Web services and the service-oriented architecture (SOA) 48
3.1.1 Understanding services 48
3.1.2 XML Web services 49
3.1.3 Service-oriented architecture (SOA) 50
3.1.4 Web service roles 53
3.1.5 Web service interaction 56

3.1.6 Service models 60
3.1.7 Web service description structure 63
3.1.8 Introduction to first-generation Web services 65
3.2 Web Services Definition Language (WSDL) 66
3.2.1 Abstract interface definition 67
3.2.2 Concrete (implementation) definition 69
3.2.3 Supplementary constructs 70
3.3 Simple Object Access Protocol (SOAP) 71
3.3.1 SOAP messaging framework 73
3.3.2 SOAP message structure 76
3.4 Universal Description, Discovery, and Integration (UDDI) 80

Chapter 4
Introduction to second-generation (WS-*) Web services technologies 89

4.1 Second-generation Web services and the service-oriented enterprise (SOE) 90
4.1.1 Problems solved by second-generation specifications 92
4.1.2 The second-generation landscape 94
4.2 WS-Coordination and WS-Transaction 96
4.2.1 Concepts 96
4.2.2 Syntax 99
4.3 Business Process Execution Language for Web Services (BPEL4WS) 100
4.3.1 Recent business process specifications 100
4.3.2 Concepts 100
4.3.3 Syntax 106
4.4 WS-Security and the Web services security specifications 109
4.4.1 General security concepts 110
4.4.2 Specifications 111
4.4.3 XML Key Management (XKMS) 112
4.4.4 Extensible Access Control Markup Language (XACML) and Extensible Rights Markup Language (XrML) 112
4.4.5 Security Assertion Markup Language (SAML) and .NET Passport 112
4.4.6 XML-Encryption and XML-Digital Signatures 113

4.4.7 Secure Sockets Layer (SSL) 113
4.4.8 The WS-Security framework 115
4.4.9 Concepts and syntax 117
4.5 WS-ReliableMessaging 118
4.5.1 WS-Addressing 119
4.5.2 Concepts 119
4.5.3 Acknowledgements 121
4.5.4 Syntax 123
4.6 WS-Policy 125
4.6.1 Concepts 126
4.6.2 Syntax 126
4.7 WS-Attachments 127

Part II
Integrating technology 131

Chapter 5
Integrating XML into applications 133

5.1 Strategies for integrating XML data representation 135
5.1.1 Positioning XML data representation in your architecture 135
5.1.2 Think "tree" (a new way of representing data) 138
5.1.3 Easy now… (don't rush the XML document model) 139
5.1.4 Design with foresight 140
5.1.5 Focus on extensibility and reusability 142
5.1.6 Lose weight while modeling! (keeping your documents trim) 142
5.1.7 Naming element-types: performance vs. legibility 143
5.1.8 Applying XML consistently 144
5.1.9 Choosing the right API (DOM vs. SAX vs. Data Binding) 145
5.1.10 Securing XML documents 147
5.1.11 Pick the right tools 148
5.1.12 Don't try this at home (fringe optimization strategies) 150
5.2 Strategies for integrating XML data validation 151
5.2.1 XSD schemas or DTDs? 151
5.2.2 Positioning DTDs in your architecture 155
5.2.3 Positioning XSD schemas in your architecture 156

5.2.4 Understand the syntactical limitations of XSD schemas 158
5.2.5 Understand the performance limitations of XSD schemas 160
5.2.6 Other fish in the sea (more schema definition languages) 160
5.2.7 Supplementing XSD schema validation 162
5.2.8 Integrating XML validation into a distributed architecture 163
5.2.9 Avoiding over-validation 165
5.2.10 Consider targeted validation 166
5.2.11 Building modular and extensible XSD schemas 167
5.2.12 Understand the integration limitations of your database 169
5.3 Strategies for integrating XML schema administration 170
5.3.1 XML schemas and the silent disparity pattern 170
5.3.2 A step-by-step XWIF process 171
5.4 Strategies for integrating XML transformation 174
5.4.1 Positioning XSLT in your architecture 174
5.4.2 Pre-transform for static caching 177
5.4.3 Create dynamic XSLT style sheets 178
5.4.4 Simplify aesthetic transformation with CSS 178
5.4.5 Understand the scalability limitations of XSLT 178
5.4.6 Strategic redundancy 179
5.5 Strategies for integrating XML data querying 179
5.5.1 Positioning XQuery in your architecture 180
5.5.2 Multi-data source abstraction 180
5.5.3 Establishing a data policy management layer 182
5.5.4 Unifying documents and data 183

Chapter 6
Integrating Web services into applications 187

6.1 Service models 188
6.1.1 Utility services 189
6.1.2 Business services 191
6.1.3 Controller services 191
6.2 Modeling service-oriented component classes and Web service interfaces 194
6.2.1 Designing service-oriented component classes (a step-by-step XWIF process) 195
6.2.2 Designing Web service interfaces (a step-by-step XWIF process) 206

6.3 Strategies for integrating service-oriented encapsulation 214
6.3.1 Define criteria for consistent logic encapsulation and interface granularity 215
6.3.2 Establish a standard naming convention 215
6.3.3 Parameter-driven vs. operation-oriented interfaces 215
6.3.4 Designing for diverse granularity 216
6.3.5 Utilize generic services consistently 217
6.3.6 Establish separate standards for internal and external services 218
6.3.7 Considering third-party Web services 219
6.4 Strategies for integrating service assemblies 220
6.4.1 Everything in moderation, including service assemblies 221
6.4.2 Modeling service assemblies 221
6.4.3 Compound service assemblies 224
6.5 Strategies for enhancing service functionality 225
6.5.1 Outputting user-interface information 225
6.5.2 Caching more than textual data 226
6.5.3 Streamlining the service design with usage patterns 227
6.6 Strategies for integrating SOAP messaging 228
6.6.1 SOAP message performance management 228
6.6.2 SOAP message compression techniques 228
6.6.3 Security issues with SOAP messaging 230
6.6.4 SOAP data types 231
6.6.5 Easing into SOAP 232

Chapter 7

Integrating XML and databases 235

7.1 Comparing XML and relational databases 236
7.1.1 Data storage and security 237
7.1.2 Data representation 237
7.1.3 Data integrity and validation 238
7.1.4 Data querying and indexing 238
7.1.5 Additional features 238
7.2 Integration architectures for XML and relational databases 239
7.2.1 Storing XML documents as database records 242
7.2.2 Storing XML document constructs as database records 244

7.2.3 Using XML to represent a view of database queries 246
7.2.4 Using XML to represent a view of a relational data model.............. 247
7.2.5 Using XML to represent relational data within an in-memory database (IMDB) 248
7.3 Strategies for integrating XML with relational databases 249
7.3.1 Target only the data you need 250
7.3.2 Avoiding relationships by creating specialized data views.............. 251
7.3.3 Create XML-friendly database models.. 251
7.3.4 Extending the schema model with annotations................................ 252
7.3.5 Non-XML data models in XML schemas... 253
7.3.6 Developing a caching strategy 253
7.3.7 Querying the XSD schema 254
7.3.8 Control XML output with XSLT.. 254
7.3.9 Integrate XML with query limitations in mind 255
7.3.10 Is a text file a legitimate repository?... 256
7.3.11 Loose coupling and developer skill sets 256
7.4 Techniques for mapping XML to relational data............................. 257
7.4.1 Mapping XML documents to relational data..................................... 257
7.4.2 The Bear Sightings application 258
7.4.3 Intrinsic one-to-one and one-to-many relationships with XML 258
7.4.4 Mapping XML to relational data with DTDs...................................... 260
7.4.5 Mapping XML to relational data with XSD schemas 267
7.5 Database extensions... 273
7.5.1 Proprietary extensions to SQL 273
7.5.2 Proprietary versions of XML specifications 274
7.5.3 Proprietary XML-to-database mapping 274
7.5.4 XML output format 274
7.5.5 Stored procedures 275
7.5.6 Importing and exporting XML documents 275
7.5.7 Encapsulating proprietary database extensions within Web services 276
7.6 Native XML databases 276
7.6.1 Storage of document-centric data... 276
7.6.2 Integrated XML schema models 277
7.6.3 Queries and data retrieval... 277
7.6.4 Native XML databases for intermediary storage.............................. 278

Part III

Integrating applications

Integrating applications 281

Chapter 8

The mechanics of application integration

The mechanics of application integration 283

8.1 Understanding application integration 284
8.1.1 Types of integration projects 284
8.1.2 Typical integration requirements 284
8.1.3 Progress versus impact 285
8.1.4 Types of integration solutions 286
8.2 Integration levels 288
8.2.1 Data-level integration 289
8.2.2 Application-level integration 290
8.2.3 Process-level integration 291
8.2.4 Service-oriented integration 292
8.3 A guide to middleware 293
8.3.1 "EAI" versus "middleware" 293
8.3.2 Shredding the Oreo 293
8.3.3 Common middleware services and products 294
8.3.4 A checklist for buying middleware 296
8.4 Choosing an integration path 300
8.4.1 Two paths, one destination 301
8.4.2 Moving to EAI 301
8.4.3 Common myths 301
8.4.4 The impact of an upgrade 302
8.4.5 Weighing your options 303

Chapter 9

Service-oriented architectures for legacy integration

Service-oriented architectures for legacy integration 305

9.1 Service models for application integration 306
9.1.1 Proxy services 307
9.1.2 Wrapper services 309
9.1.3 Coordination services (for atomic transactions) 310

9.2 Fundamental integration components 312
9.2.1 Adapters 312
9.2.2 Intermediaries 314
9.2.3 Interceptors 316
9.3 Web services and one-way integration architectures 316
9.3.1 Batch export and import 317
9.3.2 Direct data access 321
9.4 Web services and point-to-point architectures 326
9.4.1 Tightly coupled integration between homogenous legacy applications 326
9.4.2 Tightly coupled integration between heterogeneous applications 327
9.4.3 Integration between homogenous component-based applications 334
9.4.4 Integration between heterogeneous component-based applications 338
9.5 Web services and centralized database architectures 342
9.5.1 Traditional architecture 342
9.5.2 Using a Web service as a data access controller 343
9.6 Service-oriented analysis for legacy architectures 346

Chapter 10
Service-oriented architectures for enterprise integration 355

10.1 Service models for enterprise integration architectures 356
10.1.1 Process services 356
10.1.2 Coordination services (for business activities) 358
10.2 Fundamental enterprise integration architecture components 360
10.2.1 Broker 362
10.2.2 Orchestration 365
10.3 Web services and enterprise integration architectures 370
10.4 Hub and spoke 371
10.4.1 Traditional architecture 372
10.4.2 Adding integration layers with Web services 374
10.5 Messaging bus 374
10.5.1 Traditional architecture 374
10.5.2 Messaging bus solutions that utilize Web services 376
10.6 Enterprise Service Bus (ESB) 377

Chapter 11

Service-oriented integration strategies 381

11.1 Strategies for streamlining integration endpoint interfaces 383
11.1.1 Make interfaces more generic 383
11.1.2 Consolidate legacy interfaces 384
11.1.3 Consolidate proxy interfaces 385
11.1.4 Supplement legacy logic with external logic 387
11.1.5 Add support for multiple data output formats 389
11.1.6 Provide alternative interfaces for different SOAP clients 389
11.2 Strategies for optimizing integration endpoint services 391
11.2.1 Minimize the use of service intermediaries 391
11.2.2 Consider using service interceptors 391
11.2.3 Data processing delegation 393
11.2.4 Caching the provider WSDL definition 394
11.3 Strategies for integrating legacy architectures 396
11.3.1 Create a transition architecture by adding partial integration layers 396
11.3.2 Data caching with an IMDB 396
11.3.3 Utilizing a queue to counter scalability demands 397
11.3.4 Adding a mini-hub 399
11.3.5 Abstract legacy adapter technology 400
11.3.6 Leveraging legacy integration architectures 400
11.3.7 Appending Web services to legacy integration architectures 402
11.4 Strategies for enterprise solution integration 403
11.4.1 Pragmatic service-oriented integration 404
11.4.2 Integrating disparate EAI products 405
11.4.3 Respect your elders (building EAI around your legacy environments) 406
11.4.4 Build a private service registry 408
11.5 Strategies for integrating Web services security 408
11.5.1 Learn about the Web services security specifications 409
11.5.2 Build services with a standardized service-oriented security (SOS) model 409
11.5.3 Create a security services layer 409
11.5.4 Beware remote third-party services 411
11.5.5 Prepare for the performance impact 411
11.5.6 Define an appropriate system for single sign-on 412

11.5.7 Don't over-describe your services 412
11.5.8 Fortify or retreat integrated legacy systems 413
11.5.9 Take advantage of granular security 414
11.5.10 Web services and port 80 415
11.5.11 SOAP attachments and viruses 415
11.5.12 Consider the development of security policies 416
11.5.13 Don't wait to think about administration 416

Part IV
Integrating the enterprise 419

Chapter 12
Thirty best practices for integrating XML 421

12.1 Best practices for planning XML migration projects 422
12.1.1 Understand what you are getting yourself into 422
12.1.2 Assess the technical impact 424
12.1.3 Invest in an XML impact analysis 426
12.1.4 Assess the organizational impact 427
12.1.5 Targeting legacy data 428
12.2 Best practices for knowledge management within XML projects 431
12.2.1 Always relate XML to data 431
12.2.2 Determine the extent of education required by your organization 432
12.2.3 Customize a training plan 432
12.2.4 Incorporate mentoring into development projects 435
12.3 Best practices for standardizing XML applications 436
12.3.1 Incorporate standards 436
12.3.2 Standardize, but don't over-standardize 437
12.3.3 Define a schema management strategy 438
12.3.4 Use XML to standardize data access logic 440
12.3.5 Evaluate tools prior to integration 441
12.4 Best practices for designing XML applications 441
12.4.1 Develop a system for knowledge distribution 441
12.4.2 Remember what the "X" stands for 443

12.4.3 Design with service-oriented principles (even if not using Web services) 443
12.4.4 Strive for a balanced integration strategy 444
12.4.5 Understand the roles of supplementary XML technologies 445
12.4.6 Adapt to new technology developments 446

Chapter 13

Thirty best practices for integrating Web services 449

13.1 Best practices for planning service-oriented projects 450
13.1.1 Know when to use Web services 450
13.1.2 Know how to use Web services 451
13.1.3 Know when to avoid Web services 451
13.1.4 Moving forward with a transition architecture 452
13.1.5 Leverage the legacy 452
13.1.6 Sorry, no refunds (Web services and your bottom line) 453
13.1.7 Align ROIs with migration strategies 454
13.1.8 Build toward a future state 455
13.2 Best practices for standardizing Web services 456
13.2.1 Incorporate standards 456
13.2.2 Label the infrastructure 457
13.2.3 Design against an interface (not vice versa) 458
13.2.4 Service interface designer 460
13.2.5 Categorize your services 460
13.3 Best practices for designing service-oriented environments 461
13.3.1 Use SOAs to streamline business models 461
13.3.2 Research the state of second-generation specifications 461
13.3.3 Strategically position second-generation specifications 462
13.3.4 Understand the limitations of your platform 462
13.3.5 Use abstraction to protect legacy endpoints from change 463
13.3.6 Build around a security model 464
13.4 Best practices for managing service-oriented development projects 467
13.4.1 Organizing development resources 467
13.4.2 Don't underestimate training for developers 468
13.5 Best practices for implementing Web services 469
13.5.1 Use a private service registry 469

13.5.2 Prepare for administration 471
13.5.3 Monitor and respond to changes in the service hosting environments 472
13.5.4 Test for the unknown 473

Chapter 14

Building the service-oriented enterprise (SOE) 475

14.1 SOA modeling basics 476
14.1.1 Activities 478
14.1.2 Services 479
14.1.3 Processes 479
14.2 SOE building blocks 481
14.2.1 SOE business modeling building blocks 482
14.2.2 SOE technology architecture building blocks 489
14.2.3 Service-oriented security model 498
14.3 SOE migration strategy 500
14.3.1 Overview of the Layered Scope Model (LSM) 500
14.3.2 Intrinsic Layer 503
14.3.3 Internal layer 505
14.3.4 A2A layer 508
14.3.5 EAI layer 511
14.3.6 Enterprise layer 514
14.3.7 The extended enterprise 515
14.3.8 Customizing the LSM 515
14.3.9 Alternatives to the LSM 517

About the Author 519

About the Photographs 521

Index 523