INDEX

ABox. See Assertional box (ABox)
Abstract Factory pattern, 101–103
compared to TwoUse-based solution, 108
drawbacks, 103–104
Abstraction
expressiveness vs., 145
software engineering and raising level of, 9
Abstract syntax, 14, 15, 47
TwoUse, 68, 70, 119–120
ActiveRDF, 170
Adaptee classes, 71
Adapter classes, 71
ADD, 182
addAttribute, implementing in class
TUClassAdapter, 72–73
Adoption, of Selector pattern, 111
AgentRole ontology design pattern, 172
AGG, 17
Agogo, 156, 158, 188
analysis of ontology API specifications, 167–169
architecture of, 166–167
concrete syntax by example, 163–166
correlating requirements with quality attributes,
169
implementation, 166–167
key concepts, 161–163
metamodel constraints, 162–163
snippet, 162
Aligning, 52
Alignments, compute, 208
Analysis plug-ins, 113
Analyzability, 121
Annotations, 182
APIs. See Application programming interfaces (APIs)
Applicability, Selector pattern, 110
Application programming interfaces (APIs), 156.
See also Ontology APIs
OWL, 34
Approximative transformations, 52
Artifacts
OWL constructs and, 7
semantic web approaches for transforming,
134–135
semantic web technologies and, 112
TwoUse, 132–133
Artificial intelligence, ontologies and, 21
ASK, 40
Assertional box (ABox), 24, 31, 173
Assertions
semantics of, 28
syntax of, 26
ATL. See Atlas Transformation Language (ATL)
Atlas Transformation Language (ATL), 17, 52,
91, 146, 154
model transformations using, 167
ontology translation platform and, 149–150
Atlas Transformation Language (ATL)
metamodel, 146–147
limitations of, 153
Attributes, 13
Axiomatic semantics, 47
Axioms
explain, 209
templates for generating, 177
Bidirectional transformations, 52
Bindings, 180–181
Black box method for single justification, 33
BPMN. See Business Process Modeling Notation (BPMN)
Browse, 208–209
Built-ins, 144
Business Process Modeling Notation (BPMN),
15, 126, 127
Business Process Modeling Notation (BPMN)
metamodel
model extension, 128–129
Case studies
automatic generation of ontology APIs,
158–160
enabling linked data capabilities for MOF
compliant models, 124–135
improving software design patterns with OWL,
100–104
model-driven specification of ontology
translations, 142–145
modeling ontology-based information systems,
113–117
using templates in OWL ontologies, 172–174
Changeability, 122
Change management, 160
CLASS, in agogo, 163
Class-based modeling languages, 10, 12
Class Configuration, 104, 107, 108
Class description diagram
  NeOn OWL metamodel, 36, 37
  OMG OWL metamodel, 35
  OWL 2 metamodel, 38
Class descriptions, 114
Class diagrams, 100, 101. See also Unified Modeling Language (UML) class diagram
  comparing UML, OWL-DL, OWL 2, and DL-Lite, 46
Class(es), 13
  in agogo, 161, 163
  templates of, 174–175
TwoUse metamodel, 70
Class expression
  semantics of, 29
  syntax of, 26
Class expression axioms
  semantics of, 27
  syntax of, 25
Class identifier, OWL and, 116
Classifier, 13
Class Ontology, 176
Class structure alignment method, 133
Class TaskController, 100–101, 103, 104, 107, 108
Client class, 99, 100
Clients, in software design patterns, 100–101, 103
Closed-domain assumption, 30
Closed-domain template, 174
Closed-world assumption (CWA), 29–31
  using OCL over UML class-based modeling with, 79
  using SPARQL over OWL with, 79
Closure, 51
Collaborations, Selector pattern, 107–108, 109
COMM. See Core Ontology for Multimedia (COMM)
Component diagram, from TwoUse toolkit, 127
Composition, 53
Compute alignments, 208
Computing all justifications, 33
Concept
  Selector pattern, 109
  in software design patterns, 100, 101, 103
  transformations of, 144
Conceptual hierarchies, OWL and, 3–4
Concrete modeling, OWL for, 104–105
Concrete factories, 107
Concrete syntax, 14, 47
  agogo, 163–166
  EBNF definition of, 191–192
  MBOTL, 145–146
  SPARQLAS, 80–81
  TwoUse, 68, 118–119
Configuration object, 103
Conjunctive query, 31, 32
Consistency, 31
Constraints, 13
CONSTRUCT, 40
Context
  in Selector pattern, 109
  in software design patterns, 100, 101
Core Ontology for Multimedia (COMM), 113–114, 116, 118, 122
  templates and, 181–182
Core Ontology for Multimedia (COMM) APIs, 156, 158–159, 160
  comparison of size between agogo and, 168
Core ontology services, 88
Core services, 88–89
  TwoUse toolkit, 91
C-OWL, 154
Create, in COMM API, 159
Create, read, update, and delete operations (CRUD), 159, 161, 165
CRUD. See Create, read, update, and delete operations (CRUD)
CWA. See Closed-world assumption (CWA)
Data model, MMTS+OTS, 49–50
Data property axioms
  semantics of, 28
  syntax of, 25
Data ranges, syntax of, 27
Datatype, 13
  transformations of, 144
  TwoUse metamodel, 70
Debugging, 90
Debugging ontologies, 33
Debugging support, 160
Delete, in COMM API, 159
Derivation rules, 51
DESCRIBE, 40
Description logics, 22, 49, 100
  semantics of, 27–29
Description logic safe rules, 34
Description logics dialects, xxi
Description logic SROIQ(D), OWL 2 and, 27
Description logics technical space, 41–43
Description logic syntax
  E-shop with, 23
  KAT domain specification and, 116–117
  OWL 2 and, 25–26
Design integrated Ecore model, 207
Design integrated models, 206
Design integrated UML class diagram, 206
Design ontology API, 210
Design ontology engineering services, 209–210
Design ontology template, 210
Design ontology translation, 210
Diagrammatic syntax, 15
Directionality, 52
Disjunction, 32
DL-Lite class diagram, 46
DL-safe rules, 52
Domain analysis, 5
  of MMTS+OTS, 46–53
Domain closure, 172, 173
Domain concepts, core ontologies and emphasis on, 159
Domain design, by UML class diagram using UML profile for OWL, 105
Domain models, 85
Domain-specific language (DSL), 3, 161
Domain-specific language (DSL) designers and users, 188
Domain specific modeling, for ontology engineering services, 67, 70
Domain-specific modeling languages (DSML), 13, 14
DSL. See Domain-specific language (DSL)
DSML. See Domain-specific modeling languages (DSML)
EAttribute, 13
EBNF definition of concrete syntax for TwoUse, 191–192
EBNF grammar of SPARQL AS functional syntax, 192–197
of SPARQL AS Manchester syntax, 197–202
EBNF technical space, description logics technical space and, 42, 43
Eclipse Modeling Framework, 13
Eclipse Rich Client Platform, 92
Ecore, 13, 86
classes/structure of, 13, 14
defining agogo and, 167
design integrated Ecore model, 207
mapping, 131
translation rules to OWL, 204–205
Ecore Metamodel/Model, 70–72
linking with OWL, 129–130, 133
transformations of, 132–133
transforming into OWL TBox/ABox-OWLizer, 130–131
EDatatype, 13
Element identifiers, transformations of, 144
EMOF. See Essential constructs of MOF (EMOF)
Engineering services, 89–90
EPackage, 13
ERefERENCE, 13
E-Shop, 11
with description logic syntax, 23
TwoUse approach and, 68
UML class diagram profile with UML profile for OWL and TwoUse profile, 74
E-Shop domain closing with OWL axioms, 30
integrating UML and OWL in, 117–118
Essential constructs of MOF (EMOF), 12–13, 44, 45, 127
classes of, 13
Event-Model-F, APIs for, 158, 159
Event-Model-F ontology, 181
Existing metamodelling frameworks, 127
Explain axioms, 209
Explanation, 33, 91
Explanation Service, TwoUse toolkit, 92, 93
Expressiveness, abstraction vs., 145
Extended BNF, 47
Extensibility, 123
Extensible Markup Language (XML), 13, 15, 86
as serialization syntax, 47
Extension, 90
Extensional techniques, 34
Extension test, 50
Face detectors, 113
Feature diagram, 47
Feature model, 47
configuration for model checking, 55
configuration for model enrichment, 57
configuration for ontology modeling, 59
snippet of MMTS+OTS, 48
First-order logic (FOL), 15, 49
First-order logic (FOL) expressiveness, 154
Flexibility, Selector pattern and, 110
F-logic rules, 51
FOL. See First-order logic (FOL) expressiveness
Formalism, MMTS+OTS, 49
Fragment description logics, 15
Frame logic, 49
Front-end, 90
TwoUse toolkit, 92–93
Functional Syntax, 24
General-purpose modeling languages (GPML), 13
Generic architecture for MDE and ontology engineering, 87–90
Generic model-driven architecture, 90–93
GIF, 113
GPML. See General-purpose modeling languages (GPML)
Graph, 50
Graph-based syntactic transformation, 52
Graphical editor, 90
Graphical notation, 47
GReTL, 17
Gröner, Gerd, xxii
Hitting Set Tree (HST), 33
Horn rules, 49, 50
HST. See Hitting Set Tree (HST)
HTML. See HyperText Markup Language (HTML)
HTTP, 189
Human-Usable Textual Notation (HUTN), 15
HUTN. See Human-Usable Textual Notation (HUTN)
Hybrid diagram, 74
Hybrid syntactic transformations, 52
HyperText Markup Language (HTML), 14, 15
Implementation, of ontology translations, 149–150
IMPORT, in agogo, 162, 163
Inference support, 51
Inferred class hierarchy, TwoUse toolkit, 92, 93
Inferred knowledge, 144
Instance, TwoUse metamodel, 70
Instance checking, 31
Instance-schemametaschema dimension (ISM-dimension), 15
Integrated approaches. See also MOF Technical Space (MMTS)+OTS
Integrated models
classifying, 58–59
querying, 82–84
query languages for, 78–85
Integration, 52
Integrity constraints, 51
Interface, 182
Internal consistency, 54
Internal structure comparison, 34
ISM-dimension. See Instance-schemametaschema dimension (ISM-dimension)
ISO 9126, 121, 167
Java, 43, 151, 152
core services and, 91
MBOTL translated to, 148, 149
model, 11
as target language for ontology translation, 149
Jena API, 170
Jena framework, 149, 151–152
JPEG, 113
Justification
black box method for single, 33
computing all, 33
Karlsruhe ontology, 142
KAT. See K-Space Annotation Tool (KAT)
Knowledge engineering, 21
K-Space Annotation Tool (KAT), 113–114, 116
evaluation of SPARQLAS expressions according to, 121
modeling using textual language, 119
snapshot of, 122
specifying with description logic syntax, 116–117
UML class diagram of, 118
K-Space Annotation Tool (KAT) model, 120
Language dependency, 182
Languages. See Modeling languages; Query languages
Layered architecture, 12
Lexical layer, 141, 142, 144
translation problems at, 146, 153
Lexical notation, 47
Lexical translations, 150–152, 154
Linguistic metamodeling, integrating ontological metamodeling with, 189
Linked data capabilities, enabling for MOF compliant models, 124–135
case study, 125–128
related work, 134–135
TwoUse approach, 128–132
validation, 132–134
Linked data in software engineering, 189
Linked Data principles, 189
Linked open data (LOD), 124
LOD. See Linked open data (LOD)
Logical consistency, 50
Logical implication, 50
Logical rules, 51
Logic programming-based ontologies, 22
Machine learning-based classifiers, 113
Macros, use in ontologies, 182
MAFRA, 154
Maintainability, 121–122
agogo and, 168–169
Manchester OWL syntax, 142
Manchester Syntax, 24
Mapping, 53
between two models Ma and Mb, 56–57
between the UML profile for OWL and TwoUse metamodel, 74–75
Matching, 90, 131, 133
ontology, 34
MBOTL. See Model-Based Ontology Translation Language (MBOTL)
MDA. See Model-Driven Architecture (MDA)
MDE. See Model-driven engineering (MDE)
Mediation, MMTS+OTS, 52–54
Medical informatics, 66
Megamodel, 9
core services and, 91
MBOTL, 148, 149
MOF, 146
OCL, 146, 147
for ontology technologies, 35–41
for ontology templates, 176
OWL, 35–39, 146
SPARQL, 40–41
SPARQLAS, 202–203
Meta Object Facility (MOF), 10, 12, 15, 90
esential constructs of (EMOF), 12–13
Meta Object Facility (MOF)-based metamodel, 68
Meta Object Facility (MOF) compliant models, enabling linked data capabilities to,
case study, 125–128
related work, 134–135
TwoUse approach, 128–132
validation, 132–134
Meta Object Facility (MOF) metamodels for ontology technologies, 43
Meta Object Facility (MOF) model, 124–125
Meta Object Facility (MOF) Technical Space, 19, 43
MMTS. See MOF Technical Space (MMTS)
M3-neural infrastructure, 59
Model adaptation, 71–72
Model-Based Ontology Translation Language (MBOTL), 142, 145–150
screen shot, 150
semantic, syntactic, and lexical translations with, 150–152
Model-Driven Architecture (MDA), 10
Model-Driven Architecture (MDA) family of modeling languages, 58
Model-driven engineering (MDE), 3–5, 44
automatic generation of ontology APIs, 157 challenges facing, 3
Ecore, 13
extended languages for, 69
generic architecture for, 87–90
integrating ontology techniques with, 86
integrating ontology technologies with, 66–67
metamodels, 11–13
modeling languages, 13–16
models, 11
model transformations, 17
objectives of, 4
of ontology alignments, 142
querying patterns for, 189
query languages for, 17–18
requirements for linked-open data environment in, 127–128
services for, 89–90, 91–92
structure of, 9–18
technical spaces, 19
TwoUse and, 6–7, 8
use of ontology services in, 66–67
Model-driven engineering (MDE) ontologies, classification of approaches, 187
Model-driven engineering (MDE) techniques, xxiii
use in OWL ontology engineering, 67, 70
Model-driven software development, xxi, xxiii
Model-driven specification of ontology translations, 141–155
analysis, 153
case study, 142–145
e samples, 150–153
related work, 154–155
TwoUse approach, 145–150
Model enrichment, MMTS+OTS and, 56–57
Model extension, 128–130
Modeling design patterns, 86
Modeling languages, 13–16
abstract syntax, 15
extending into OWL, 128–132
semantics, 15, 16
syntax, 15, 16
transformation into OWL, 188
Modeling level, 53
Model libraries, 148
Model (M1), 11, 53. See also Megamodel; Metamodel (M2)
Model management services, 88–89
Model-theoretic semantics, 47
Model transformations, 17, 91, 130–131
Model validation, MMTS+OTS and, 54–56
MOF. See under Meta Object Facility (MOF)
MOF Technical Space (MMTS), 44
compared to OTS, 45
MOF Technical Space (MMTS)+OTS, 44–45, 59
commonalities and variations, 46–53
data model, 49–50
formalism, 49
language, 47–49
mediation, 52–53
model enrichment, 56–57
modeling level, 53
model validation, 54–56
ontology modeling, 58
querying, 51
reasoning, 50
rules, 51–52
similarities between OWL and UML class-based modeling, 45–46
snippet of feature model, 48
transformation, 52
M3O ontology, 181
MOST (Marrying Ontology and Software Technologies), xxii
MPEG, 113
Multiple MOF models, 128
NAF. See Negation as failure (NAF)
Name assumption, 27–31
Natural-language processing, 21
NCSSs, See Non-commenting source statements (NCSSs)
Negation, 32
Negation as failure (NAF), 30
NeOn OWL metamodel, 36–38
NeOn Toolkit, 182
NExpTime-Complete (OWL DL), 24
NLogSpace-Complete (OWL 2 QL), 24
Non-commenting source statements (NCSSs), 160, 168
Notations for templates in OWL, 179–180
OBA. See Ontology-based architecture (OBA)
Object Adapter Pattern, 70–71
Object-based data model, 50
Object Constraint Language (OCL), 17–18, 51, 65, 86, 90
extension of, 146
semantics of, 18
in strategy pattern, 100–101
UML class-based software development and, 114–115
using over OWA and UML class-based modeling with SPARQL, 79–80
using over UML class-based modeling with CWA, 79
Object Constraint Language (OCL) expressions, 145
Object Constraint Language (OCL) library, 148
Object Constraint Language (OCL) metamodel, 146, 147
Object Management Group (OMG), 124
  definition of model transformation language, 17
  MOF Query/View/Transformation, 146
  Ontology Definition Metamodel, 35, 135
  OWL metamodel, 35–36
  Query/View/Transformation (QVT), 17, 52, 146, 154
  RDF Metamodel, 35
  Request for Proposal for MOF to RDF Structural Mapping, 135
  UML Profile for OWL, 175, 179, 180
Object-oriented (OO) representations, 156
Object property axioms
  semantics of, 28
  syntax of, 25
Object recognizers, 113
OCL. See Object Constraint Language (OCL)
ODD. See Ontology-driven development (ODD)
ODEDialect, 154–155
ODPs. See Ontology design patterns (ODPs)
OEA. See Ontology-enabled architecture (OEA)
OED. See Ontology-enabled development (OED)
OMG. See Object Management Group (OMG)
Ontological metamodeling, integrating with linguistic metamodeling, 189
Ontological reasoning technology, scalability of, 190
Ontological Technical Space (OTS), 41–43, 44.
  See also MOF Technical Space (MMTS)+OTS
  compared to MMTS, 45
Ontology, 3–5
  debugging, 33
  defined, 22
  ontology modeling, 22–23
  role of, 21
  as templateable elements, 176
  templates and versions of, 181
Ontology Alignment Evaluation Initiative, 142
Ontology APIs, 5
  automatic generation of, 8, 156–170, 188
  analysis, 167–169
  case study, 158–160
  limitations of current approaches, 157
  related work, 169–170
  TwoUse approach, 161–167
  design, 210
  specification of, 89
Ontology-based architecture (OBA), 59
Ontology-based domain specific languages, 188
Ontology-Based Information Services, 31–33
  querying, 31–33
  reasoning services, 31
Ontology-based information systems, modeling, 112–123
  case study, 113–117
  limitations, 123
  ontology-based software development, 116–117
  TwoUse approach, 117–121
  UML class-based software development, 113–115
  validation, 121–123
  Ontology-based software design patterns, 187
  Ontology-based software development, 116–117
  Ontology Definition Metamodel, 58
  Ontology design patterns (ODPs), 156, 171, 172
  in agogo, 161
  templates and, 181–182
ontologic endabled architecture (OEA), 59
Ontology modeling, 8, 89
  MDE support for, 67, 70
  MMTS+OTS and, 58, 59
Ontology Representation Language, 58
Ontology techniques, integrating MDE with, 86
Ontology technologies, xx1, 4
  integrating with model-driven engineering, 66–67
  metamodels for, 35–41
  OWL, 35–39
  SPARQL, 40–41
  model-driven viewpoint of, 42
  TwoUse and, 6
Ontology templates. See also Templates
  defined, 176
  design, 210
Ontology translation, 89
  design, 210
  model-driven specification of, 141–155
  analysis, 153
  case study, 142–145
  examples, 150–153
  related work, 154–155
  TwoUse approach, 145–150
  process, 148–149
  satisfying requirements of, 153
  specifying, 188
Ontology translation languages, 4–5
Ontology web languages, 3. See also Web Ontology Language (OWL)
OntoMap, 154
OntoMorph, 154
OO. See Object-oriented (OO) representations
Open-world assumption (OWA), 29–31, 173
using OCL and SPARQL over, 79–80
using SPARQL over OWL with, 79
Operation, 13
in agogo, 162
defined using SPARQL update syntax, 165
Operation call expressions, 148
OTS. See Ontological Technical Space (OTS)
OWA. See Open-world assumption (OWA)
OWL. See under Web Ontology Language (OWL)

Package, 13
templates of, 174–175
TwoUse metamodel, 70
PACKAGE, in agogo, 163
ParameterElement, 177
Parameters, 13
Participants, Selector pattern, 107–108, 109
Pattern
in agogo, 161
defining, 164
as first-class citizen, 160, 164
mapping property onto, 165, 166
PIM. See Platform-independent model (PIM)
Platform-independent model (PIM), 10, 167
Platform-specific model (PSM), 10, 167
Plug-ins, 182
Portability, agogo and, 169
Pragmatic layer, 144
Precise transformations, 52
Predicate function, 150, 152
Profiled UML class diagram
of ontology-based solution, 106
with Strategy pattern, 106–107
Properties diagram
NeOn OWL metamodel, 36, 37
OMG OWL metamodel, 36
OWL 2 metamodel, 38, 39
Property
mapping onto pattern, 165, 166
TwoUse metamodel, 70
PROPERTY, in agogo, 163
Protégé, 182
PSM. See Platform-specific model (PSM)
PTime-Complete (OWL 2 EL, OWL 2 RL), 24

Queries, as templateable elements, 176
Query answering, 91, 128
Querying, 31–33
MMTS+OTS, 51
SPARQLAS, 131–132, 133–134
use case, 133–134
Querying ontologies, using OWL syntax, 80–82
Querying patterns, for MDE in OWL, 189
Query languages, 17–18
for integrated models, 78–85
combining existing approaches, 78–80
SPARQLAS, 80–82
SPARQLAS+TwoUse, 82–84
Query OWL ontologies, 209
Query templates, 180–181
Query UML class-based models, 209
Query/View/Transformation (QVT), 17, 52, 146, 154

RDF. See under Resource Description Framework (RDF)
RDFS, 50, 189
metamodel, 35, 36
overview, 49
RDFS-Java/OO mapping approaches, 170
RDFT, 154
RDF/XML syntax, 24
Reaction rules, 51
Read, in COMM API, 159
Realization, 31
Reasoning, 85, 91
MMTS+OTS, 50
OWL, 117
Reasoning services, 31
Recognizers, 113–114, 115, 117
Recursive algorithm, 178–179
RecursiveResult, 179
Reference ontology, 142
References, 13
Relational data model, 50
Relational structure comparison, 34
REMOVE, 182
Representation, in megamodel, 11
Resource Description Framework (RDF), 125, 189
semantic patterns in, 182
Resource Description Framework (RDF) library, 148
Resource Description Framework (RDF) models, 57
Resource Description Framework (RDF) Reactor, 170
Restriction on programming language, 161
Retrieval, 31
Reusability
agogo and, 168
of algorithm descriptions, 122–123
Reuse, Selector pattern and, 110
RQ1, 159, 161
RQ2, 128, 160, 161
RQ3, 128, 160, 161
RQ4, 160, 161
RQ5, 160, 161
Ruby programs, 170
RuleML, OWL and, 34
Rules, 34
derivation, 51
DL-safe, 52
F-logic, 51
horn, 49, 50
logical, 51
MMTS+OTS, 51
reaction, 51
Runtime instances (M0), 53
Safety, 51
SAIQL, 174
templates and, 172
Satisfiability, 31
Scalability of ontological reasoning technology, 190
Schema management, 5
SELECT, 40
Selector pattern, 100, 105–108, 187
advantages, 110–111
applicability, 110
comparison, 108
drawbacks, 110
participants and collaborations, 107–108, 109
structure, 106–107
validation, 109–111
Semantic annotation, 113–114
interface for, 182
Semantic annotation pattern, 160
ontology and API for, 158–159
Semantic layer, 141, 142, 144
translation problems at, 145–146, 153
Semantic patterns, in RDF, 182
Semantic reasoning capabilities, 154
Semantics, 14, 15
axiomatic, 47
MBOTL, 148
model-theoretic, 47
2NExpTimeComplete (OWL 2), 27–29
of OCL, 18
for SPARQL, 34
of templates, 177–179
translational, 47
of UML, 16
Semantic transformation, 52
Semantic translation, 150–152, 154
of complex structures, 152–153
Semantic web approaches, for transforming software artifacts, 134–135
Semantic Web Rule Language (SWRL), 34, 86
metamodel, 38–39
rules, 172, 173, 174
templates and, 172
UML Profile for, 175, 179, 180
Semantic web stack, 21, 22
Semantic web technologies, 124, 189
ontology-based information systems and, 112
Sequence diagram, of OWL-based solution, 108
Serialization syntax, 47
Sesame API, 170
Snoogle, 154
Software design patterns, improving with OWL, 99–111
case study, 100–104
related work, 111
TwoUse approach and, 104–108
validation, 109–111
Software engineer, UML use case diagram and, 126
Software modeling, xxi
SPARQL, 17, 32–33, 90, 189
EBNF syntax, 40
MBOTL translated to, 148, 149
metamodel, 40–41
queries, 40–41, 51, 88, 151, 164
syntax
in agogo, 164–166
CRUD operations using, 161
as target language for ontology translation, 149–150
transformation from OCL to, 145
transformation from SPARQLAS to, 81–82
update syntax, definition of operation using, 165
using over OWA and UML class-based modeling with OCL, 79–80
using over OWL with CWA, 79
using over OWL with OWA, 79
SPARQLAS, 68, 80–82
correspondence syntax, 80–81
EBNF grammar of SPARQLAS functional syntax, 192–197
evaluating expressions, 121
metamodel, 81, 202–203
ontology services and, 69
querying with, 131–132, 133–134
query templates, 180–181
transformation to SPARQL, 81–82
SPARQLAS Manchester syntax, EBNF grammar of, 197–202
SPARQLAS4TwoUse, 82–84
expressions in software design patterns, 106
positioning among existing approaches, 84, 85
query operations, 107, 207
SPARQL-like language (SPARQL-DL), 31, 32–33, 78
abstract syntax, 32–33, 174
templates and, 172
TwoUse toolkit and, 91
Speaker identifiers, 113
SQL, 17
Strategy pattern, 100–103
compared to TwoUse-based solution, 108
drawbacks, 102, 103–104
profiled UML class diagrams with, 106–107
String distance method, 133
Structure, Selector pattern, 106–107
Structure comparison techniques, 34
Subsumption, 31, 50
Support vector machines (SVM), 113, 114
SVM. See Support vector machines (SVM)
SWRL. See under Semantic Web Rule Language (SWRL)
Syntactic layer, 141, 142, 144
translation problems at, 146, 153
Syntactic transformation, 52
Syntactic translation, 150–152, 154
do complex structures, 152–153
Syntax, 15
abstract, 14, 15, 47, 68, 70
TwoUse, 119–120
concrete (See Concrete syntax)
diagrammatic, 15
2NExpTimeComplete (OWL 2), 24–26, 179
OWL, 80–82
serialization, 47
SPARQLAS functional, 192–197
SPARQLAS Manchester, 197–202
SPARQL EBNF, 40
SPARQL-like language abstract, 32–33
textual, 15
TwoUse approach, 72–77
pure UML class diagrams, 75
textual notation, 75–77
UML profile for OWL, 72–75
UML, 16
System, 53
Target classes, 71
Target platform, transformation into, 149
Tarski-style model theory of OWL, 117
TBox. See Terminological box (TBox)
Technical space, 19, 41–43, 44, 45
organization of features according to, 53
TemplateableElement, 176
Template binding, 176, 177
Template binding realization algorithm, 178
TemplateParameter, 177
TemplateParameterSubstitution, 177
Templates
advantages of, 174
as first-class citizens, 172
ontology design patterns and, 181–182
in OWL ontologies, 171–183, 188
analysis, 181–182
case study, 172–174
limitations, 182
notations for, 179–180
related work, 182–183
TwoUse approach, 174–181
query, 180–181
semantics of, 177–179
TemplateSignature, 176
Terminological box (TBox), 24, 31, 173
Testability, 122
Selector pattern and, 110
Textual editor, 90
Textual language, modeling KAT using, 119
Textual notation, 75–77
Textual syntax, 15
Traceability matrix, 69
agogo, 169
Transformation
MDE and, 10
MMTS+OTS, 52
Transformation rules, 17
for TwoUse approach, 68
Translational semantics, 47
Translation metamodel, 146, 147
Translation rules, 143
Ecore to OWL, 204–205
example, 146
Transparency, templates and, 182
Tree, 50
TUClassAdapter, 76
implementing operation addAttribute in class, 72–73
Turtle, 24
Tutorials, xxv–xxvi
2NExpTimeComplete (OWL 2), 24
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) approach, xxi, 6
applications in ontology engineering, 7–8
application to software design patterns, 104–108
OWL for conceptual modeling, 104–105
Selector pattern, 105–108
for automatic generation of ontology APIs, 161–167
development life cycle, 125–126
EBNF definition of concrete textual syntax for, 191–192
limitations of, 123
model extension, 128–130
for modeling ontology-based information systems, 117–121
model transformation, 130–131
querying with SPARQLAS, 131–132
templates in OWL ontologies, 174–182
use cases, 206–210
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) approach syntax, 72–77
pure UML class diagrams, 75
textual notation, 75–77
UML profile for OWL, 72–75
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) conceptual architecture, 65–77
metamodeling architecture, 70–72
requirements for integrating ontology technologies and model-driven engineering, 66–67
requirements of TwoUse approach, 68–70
syntax, 72–77
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) measurement, 132
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) metamodel, 69, 70–72, 73
mapping between UML profile for OWL and, 74–75
TwoUse (Transforming and Weaving Ontologies and UML for Software Engineering) toolkit, 86–93, 188
core services, 88–89
engineering services, 89–90
front-end, 90

generic architecture for MDE and ontology engineering, 87–90
instantiating generic model-driven architecture, 90–93
use case descriptions, 87
validation, 132–134
Type metamodel, 147–148
Types, 13

**UML.** See under Unified Modeling Language (UML)

**UNA.** See Unique Name Assumption (UNA)

Unidirectional transformations, 52, 54
Unified Modeling Language (UML), 3, 4, 13, 86, 90
as diagrammatic notation, 15
as metamodeling language, 11
model, 11
as modeling language, 10
overview, 47, 49
semantics of, 15, 16
structure, semantics, and syntax of, 16
templates of packages and classes, 174–175
Unified Modeling Language (UML) class-based metamodel, OWL 2 metamodel adapted for, 72, 73
Unified Modeling Language (UML) class-based modeling, 5–6
adapting OWL class for, 70–72
integrating OWL constructs in, 66, 70
integrating OWL ontologies in, 187
OWL and, 65–66
similarities with OWL modeling, 45–46
TwoUse and, 6–7
use of ontology services in, 66–67, 70
using OCL and SPARQL over, 79–80
using OCL over with CWA, 79
Unified Modeling Language (UML) class-based modeling languages, 10
integration between OWL and, 86, 87
Unified Modeling Language (UML) class-based model query, 209
Unified Modeling Language (UML) class-based software development, 113–115
Unified Modeling Language (UML) class diagram, 15, 17, 18, 50, 54
design integrated, 206
domain design by, 105
of KAT, 114, 118
pure, 75
Unified Modeling Language (UML) component diagrams, 126
Unified Modeling Language (UML) metamodel, 15
model extension, 128–129
Unified Modeling Language (UML) models, checking consistency of, 54–56
Unified Modeling Language (UML) paradigm dominance, Selector pattern and, 111
Unified Modeling Language (UML) profile, 68
ontology modeling with, 58
for OWL, 72–75
mapping between TwoUse metamodel and, 74–75
for SWRL, 175, 179, 180
Unified Modeling Language (UML) sequence diagram
of KAT algorithms, 114, 115
of strategy and abstract factory patterns, 103
Unified Modeling Language (UML) use case diagrams, 126
Unique Name Assumption (UNA), 78
Universal quantification, 32
Update, in COMM API, 159
URLs, 189
Usability
agogo and, 168
templates and, 182
Use case descriptions, 87, 88
Use case querying, 133
Use cases, 131, 132
BPMN and UML metamodels for, 128–129
of TwoUse approach, 206–210
Validation, 90
of Selector pattern, 109–111
TwoUse approach, 121–123
TwoUse toolkit, 132–134
Valid binding, 177
Values, transformations of, 144
Variable mapping, 33
Variable ?self, 83–84
Variants
in Selector pattern, 109
in software design patterns, 100–101, 103, 104, 108

Walter, Tobias, xxii

Web of models, 124
Web Ontology Language (OWL), xxiii, 3–4, 22, 24–31
for conceptual modeling, 104–105
for domain concepts in networked environments, 125
extending modeling languages into, 128–132
integration between UML class-based modeling languages and, 86, 87
linking Ecore metamodels with, 129–130, 133
modeling and querying patterns for MDE in, 189
overview, 49
RuleML and, 34
software design patterns and (See Software design patterns, improving with OWL) sub-languages, 24
syntax of, querying ontologies using, 80–82
transformation of modeling languages into, 188
transformation to, 207–208
translation rules from Ecore, 204–205
UML class-based modeling and, 5–6, 65–66
UML profile for, 72–75
using SPARQL over with CWA, 79
using SPARQL over with OWA, 79
world assumption and name assumption, 27–31
Web Ontology Language 2 (OWL 2), 90, 130
class diagram, 46
Functional Syntax, 43
graphical syntax, 179
semantics of, 27–29
syntax of, 24–26
Vocabulary $V_o$, 24–25
Web Ontology Language (OWL) application
program interface (OWL API), 34
Web Ontology Language (OWL)-based solution,
sequence diagram of, 108
Web Ontology Language (OWL) class, adapting
for UML class-based modeling, 70–72
Web Ontology Language (OWL) constructs,
integrating in UML class-based modeling,
66, 70
Web Ontology Language (OWL)-DL, class
diagram, 46
Web Ontology Language 2 (OWL 2)FS editor, 90
Web Ontology Language (OWL) Functional
Syntax, SPARQLAS query using, 80
Web Ontology Language (OWL) library, 148
Web Ontology Language (OWL) Manchester
Syntax, 76
Web Ontology Language 2 (OWL 2) Manchester
Syntax
SPARQLAS query using, 80, 81, 83
Web Ontology Language (OWL) metamodel,
35–39
extending with templates, 174–177
NeOn OWL metamodel, 36–38
OMG OWL metamodel, 35–36
SWRL, 38–39
Web Ontology Language 2 (OWL 2) metamodel,
38, 39
adapted for UML class-based metamodel, 72, 73
Web Ontology Language (OWL) modeling,
116–117
similarities with UML class-based modeling,
45–46
TwoUse and, 7
Web Ontology Language (OWL) ontologies
classification of approaches, 187
integration with UML class-based modeling,
187
query, 209
template mechanics and, 177–178
templates in (See under Templates)
Web Ontology Language (OWL) ontology
engineering, use of MDE techniques in,
67, 70
Web Ontology Language (OWL) profiled class
diagram, 105–106
Web Ontology Language 2 (OWL 2) reasoner,
91
Web Ontology Language (OWL) reasoning, 117
Web Ontology Language (OWL) reasoning
service, 117–118
Web Ontology Language (OWL) TBox/ABox-
OWLizer, 130–131
Web Ontology Language (OWL)/XML syntax,
24
Well-formedness, of language, 14
WHERE clause, SPARQL queries, 41
World assumption, 27–31, 85
World Wide Web Consortium standard Web
Ontology Language. See Web Ontology
Language (OWL)
X-COSIMO, APIs for, 156, 158, 159
XML. See Extensible Markup Language (XML)
XML Schema Datatypes library, 148