

INTERNATIONAL
STANDARD

ISO/IEC
19501

First edition
2005-04-01

**Information technology — Open
Distributed Processing — Unified
Modeling Language (UML) Version 1.4.2**

*Technologies de l'information — Traitement distribué ouvert —
Langage de modélisation unifié (UML), version 1.4.2*

Reference number
ISO/IEC 19501:2005(E)



© ISO/IEC 2005

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2005

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

1	Scope	1
2	Normative references	1
2.1	Identical Recommendations International Standards	1
3	General Information	2
3.1	Description	2
3.2	Outside the Scope of the UML	3
3.2.1	Programming Languages	3
3.2.2	Tools	3
3.2.3	Process	3
3.3	Primary Artifacts of the UML	4
3.3.1	UML-defining Artifacts	4
3.3.2	Development Project Artifacts	4
3.4	Motivation to Define the UML	5
3.4.1	Why We Model	5
3.4.2	Industry Trends in Software	5
3.4.3	Prior to Industry Convergence	6
3.5	Goals of the UML	6
3.5.1	Comparing UML to Other Modeling Languages	8
3.5.2	Features of the UML	8
3.6	UML - Past, Present, and Future	10
3.6.1	UML 0.8 - 0.91	10
3.6.2	UML Partners	11
3.6.3	UML - Present and Future	11
4	UML Semantics	13
	Part 1 - Background	
4.1	Introduction	13
4.1.1	Purpose and Scope	13
4.1.2	Approach	13
4.2	Language Architecture	14
4.2.1	Four-Layer Metamodel Architecture	14
4.2.2	Package Structure	15
4.3	Language Formalism	17
4.3.1	Levels of Formalism	17
4.3.2	Package Specification Structure	18
4.3.3	Use of a Constraint Language	19
4.3.4	Use of Natural Language	19
4.3.5	Naming Conventions and Typography	20
	Part 2 - Foundation	
4.4	Foundation Package	20
4.5	Core	21
4.5.1	Overview	21
4.5.2	Abstract Syntax	21
4.5.3	Well-Formedness Rules	51
4.5.4	Detailed Semantics	63

4.6	Extension Mechanisms	69
4.6.1	Overview	69
4.6.2	Abstract Syntax	71
4.6.3	Well-Formedness Rules	74
4.6.4	Detailed Semantics	76
4.6.5	Notes	77
4.7	Data Types	78
4.7.1	Overview	78
4.7.2	Abstract Syntax	78

Part 3 - Behavioral Elements

4.8	Behavioral Elements Package	85
4.9	Common Behavior	85
4.9.1	Overview	85
4.9.2	Abstract Syntax	85
4.9.3	Well-Formedness Rules	96
4.9.4	Detailed Semantics	101
4.10	Collaborations	103
4.10.1	Overview	103
4.10.2	Abstract Syntax	104
4.10.3	Well-Formedness Rules	111
4.10.4	Detailed Semantics	115
4.10.5	Notes	118
4.11	Use Cases	119
4.11.1	Overview	119
4.11.2	Abstract Syntax	119
4.11.3	Well-Formedness Rules	122
4.11.4	Detailed Semantics	124
4.11.5	Notes	128
4.12	State Machines	128
4.12.1	Overview	128
4.12.2	Abstract Syntax	128
4.12.3	Well-Formedness Rules	136
4.12.4	Detailed Semantics	140
4.12.5	Notes	148
4.13	Activity Graphs	152
4.13.1	Overview	152
4.13.2	Abstract Syntax	152
4.13.3	Well-Formedness Rules	156
4.13.4	Detailed Semantics	159
4.13.5	Notes	160

Part 4 - General Mechanisms

4.14	Model Management	161
4.14.1	Overview	161
4.14.2	Abstract Syntax	161
4.14.3	Well-Formedness Rules	165
4.14.4	Semantics	170
4.14.5	Notes	174

5 UML Notation Guide	177
Part 1 - Background	
5.1 Introduction	177
Part 2 - Diagram Elements	
5.2 Graphs and Their Contents	178
5.3 Drawing Paths	178
5.4 Invisible Hyperlinks and the Role of Tools	179
5.5 Background Information	179
5.5.1 Presentation Options	179
5.6 String	179
5.6.1 Semantics	179
5.6.2 Notation	179
5.6.3 Presentation Options	180
5.6.4 Examples	180
5.6.5 Mapping	180
5.7 Name	180
5.7.1 Semantics	180
5.7.2 Notation	180
5.7.3 Example	180
5.7.4 Mapping	181
5.8 Label	181
5.8.1 Semantics	181
5.8.2 Notation	181
5.8.3 Presentation Options.....	181
5.8.4 Example	181
5.9 Keywords	181
5.10 Expression	182
5.10.1 Semantics	182
5.10.2 Notation	182
5.10.3 Examples	182
5.10.4 Mapping	182
5.10.5 OCL Expressions	182
5.10.6 Selected OCL Notation	183
5.10.7 Examples	183
5.11 Note	183
5.11.1 Semantics	183
5.11.2 Notation	183
5.11.3 Presentation Options	183
5.11.4 Example	184
5.11.5 Mapping	184
5.12 Type-Instance Correspondence	184
Part 3 - Model Management	
5.13 Package	186
5.13.1 Semantics	186
5.13.2 Notation	186

5.13.3 Presentation Options.....	186
5.13.4 Style Guidelines	187
5.13.5 Example	187
5.13.6 Mapping	188
5.14 Subsystem	188
5.14.1 Semantics	188
5.14.2 Notation	188
5.14.3 Presentation Options	189
5.14.4 Example	190
5.14.5 Mapping	193
5.15 Model	193
5.15.1 Semantics	193
5.15.2 Notation	193
5.15.3 Presentation Options	193
5.15.4 Example	194
5.15.5 Mapping	194

Part 4 - General Extension Mechanisms

5.16 Constraint and Comment	195
5.16.1 Semantics	195
5.16.2 Notation	195
5.16.3 Example	196
5.16.4 Mapping	196
5.17 Element Properties	197
5.17.1 Semantics	197
5.17.2 Notation	197
5.17.3 Presentation Options	198
5.17.4 Style Guidelines	198
5.17.5 Example	198
5.17.6 Mapping	198
5.18 Stereotypes	199
5.18.1 Semantics	199
5.18.2 Notation	199
5.18.3 Examples	200
5.18.4 Mapping	200

Part 5 - Static Structure Diagrams

5.19 Class Diagram	201
5.19.1 Semantics	201
5.19.2 Notation	201
5.19.3 Mapping	201
5.20 Object Diagram	201
5.21 Classifier	201
5.22 Class	202
5.22.1 Semantics	202
5.22.2 Basic Notation	202
5.22.3 Presentation Options	202
5.22.4 Style Guidelines	203
5.22.5 Example	203

5.22.6 Mapping	203
5.23 Name Compartment	204
5.23.1 Notation	204
5.23.2 Mapping	204
5.24 List Compartment	204
5.24.1 Notation	204
5.24.2 Presentation Options.....	205
5.24.3 Example	206
5.24.4 Mapping	206
5.25 Attribute	207
5.25.1 Semantics	207
5.25.2 Notation	207
5.25.3 Presentation Options	208
5.25.4 Style Guidelines	209
5.25.5 Example	209
5.25.6 Mapping	209
5.26 Operation	209
5.26.1 Semantics	209
5.26.2 Notation	209
5.26.3 Presentation Options	210
5.26.4 Style Guidelines	211
5.26.5 Example	211
5.26.6 Mapping	211
5.27 Nested Class Declarations	212
5.27.1 Semantics	212
5.27.2 Notation	212
5.27.3 Mapping	212
5.28 Type and Implementation Class.....	212
5.28.1 Semantics	212
5.28.2 Notation	213
5.28.3 Example	213
5.28.4 Mapping	213
5.29 Interfaces	214
5.29.1 Semantics	214
5.29.2 Notation	214
5.29.3 Example	214
5.29.4 Mapping	215
5.30 Parameterized Class (Template).....	215
5.30.1 Semantics	215
5.30.2 Notation	215
5.30.3 Presentation Options.....	216
5.30.4 Example	216
5.30.5 Mapping	216
5.31 Bound Element	217
5.31.1 Semantics	217
5.31.2 Notation	217
5.31.3 Style Guidelines	217
5.31.4 Example	217
5.31.5 Mapping	217
5.32 Utility	218

5.32.1 Semantics	218
5.32.2 Notation	218
5.32.3 Example	218
5.32.4 Mapping	218
5.33 Metaclass	218
5.33.1 Semantics	218
5.33.2 Notation	218
5.33.3 Mapping	219
5.34 Enumeration	219
5.34.1 Semantics	219
5.34.2 Notation	219
5.34.3 Mapping	219
5.35 Stereotype Declaration	219
5.35.1 Semantics	219
5.35.2 Notation	219
5.35.3 Mapping	222
5.36 Powertype	222
5.36.1 Semantics	222
5.36.2 Notation	222
5.36.3 Mapping	222
5.37 Class Pathnames	223
5.37.1 Notation	223
5.37.2 Example	223
5.37.3 Mapping	223
5.38 Accessing or Importing a Package	223
5.38.1 Semantics	223
5.38.2 Notation	224
5.38.3 Example	224
5.38.4 Mapping	224
5.39 Object	225
5.39.1 Semantics	225
5.39.2 Notation	225
5.39.3 Presentation Options	225
5.39.4 Style Guidelines	226
5.39.5 Variations	226
5.39.6 Example	226
5.39.7 Mapping	226
5.40 Composite Object	226
5.40.1 Semantics	226
5.40.2 Notation	227
5.40.3 Example	227
5.40.4 Mapping	227
5.41 Association	227
5.42 Binary Association	228
5.42.1 Semantics	228
5.42.2 Notation	228
5.42.3 Presentation Options	229
5.42.4 Style Guidelines	229
5.42.5 Options	229
5.42.6 Example	229

5.42.7 Mapping	230
5.43 Association End	230
5.43.1 Semantics	230
5.43.2 Notation	230
5.43.3 Presentation Options	232
5.43.4 Style Guidelines	232
5.43.5 Example	232
5.43.6 Mapping	233
5.44 Multiplicity	233
5.44.1 Semantics	233
5.44.2 Notation	233
5.44.3 Style Guidelines	233
5.44.4 Example	233
5.44.5 Mapping	234
5.45 Qualifier	234
5.45.1 Semantics	234
5.45.2 Notation	234
5.45.3 Presentation Options	234
5.45.4 Style Guidelines	234
5.45.5 Example	235
5.45.6 Mapping	235
5.46 Association Class	235
5.46.1 Semantics	235
5.46.2 Notation	235
5.46.3 Presentation Options	235
5.46.4 Style Guidelines	235
5.46.5 Example	236
5.46.6 Mapping	236
5.47 N-ary Association	236
5.47.1 Semantics	236
5.47.2 Notation	236
5.47.3 Style Guidelines	237
5.47.4 Example	237
5.47.5 Mapping	237
5.48 Composition	237
5.48.1 Semantics	237
5.48.2 Notation	238
5.48.3 Design Guidelines	238
5.48.4 Example	239
5.48.5 Mapping	240
5.49 Link	240
5.49.1 Semantics	240
5.49.2 Notation	240
5.49.3 Example	241
5.49.4 Mapping	241
5.50 Generalization	241
5.50.1 Semantics	241
5.50.2 Notation	241
5.50.3 Presentation Options	242
5.50.4 Mapping	244

5.51	Dependency	245
5.51.1	Semantics	245
5.51.2	Notation	245
5.51.3	Presentation Options.....	246
5.51.4	Example	246
5.51.5	Mapping	247
5.52	Derived Element	247
5.52.1	Semantics	247
5.52.2	Notation	247
5.52.3	Style Guidelines	247
5.53	InstanceOf	247
5.53.1	Semantics	247
5.53.2	Notation	248
5.53.3	Mapping	248

Part 6 - Use Case Diagrams

5.54	Use Case Diagram	248
5.54.1	Semantics	248
5.54.2	Notation	248
5.54.3	Example	249
5.54.4	Mapping	249
5.55	Use Case	249
5.55.1	Semantics	249
5.55.2	Notation	250
5.55.3	Presentation Options.....	250
5.55.4	Style Guidelines	250
5.55.5	Mapping	250
5.56	Actor	250
5.56.1	Semantics	250
5.56.2	Notation	250
5.56.3	Presentation Options.....	250
5.56.4	Style Guidelines	251
5.56.5	Mapping	251
5.57	Use Case Relationships	251
5.57.1	Semantics	251
5.57.2	Notation	251
5.57.3	Example	252
5.57.4	Mapping	252
5.58	Actor Relationships	252
5.58.1	Semantics	252
5.58.2	Notation	252
5.58.3	Example	253
5.58.4	Mapping	253

Part 7 - Interaction Diagrams

5.59	Collaboration	253
5.59.1	Semantics	253
5.60	Sequence Diagram	254
5.60.1	Semantics	254

5.60.2 Notation	254
5.60.3 Presentation Options	255
5.60.4 Example	256
5.60.5 Mapping	258
5.61 Object Lifeline	260
5.61.1 Semantics	260
5.61.2 Notation	260
5.61.3 Presentation Options	260
5.61.4 Example	261
5.61.5 Mapping	261
5.62 Activation.....	261
5.62.1 Semantics	261
5.62.2 Notation	261
5.62.3 Example	262
5.62.4 Mapping	262
5.63 Message and Stimulus.....	262
5.63.1 Semantics	262
5.63.2 Notation	262
5.63.3 Presentation options	262
5.63.4 Example	264
5.63.5 Mapping	264
5.64 Transition Times.....	264
5.64.1 Semantics	264
5.64.2 Notation	264
5.64.3 Presentation Options.....	264
5.64.4 Example	264
5.64.5 Mapping	264

Part 8 - Collaboration Diagrams

5.65 Collaboration Diagram	264
5.65.1 Semantics	264
5.65.2 Notation	265
5.65.3 Example	266
5.65.4 Mapping	267
5.66 Pattern Structure	267
5.66.1 Semantics	267
5.66.2 Notation	268
5.66.3 Mapping	270
5.67 Collaboration Contents.....	270
5.67.1 Semantics	271
5.67.2 Notation.....	271
5.67.3 Mapping	272
5.68 Interactions.....	272
5.68.1 Semantics	272
5.68.2 Notation	273
5.68.3 Mapping	273
5.68.4 Example	273
5.69 Collaboration Roles	273
5.69.1 Semantics	273
5.69.2 Notation	273

5.69.3 Presentation options	274
5.69.4 Example	275
5.69.5 Mapping	275
5.70 Multiobject	275
5.70.1 Semantics	275
5.70.2 Notation	275
5.70.3 Example	276
5.70.4 Mapping	276
5.71 Active object	276
5.71.1 Semantics	276
5.71.2 Notation	276
5.71.3 Example	277
5.71.4 Mapping	277
5.72 Message and Stimulus	277
5.72.1 Semantics	277
5.72.2 Notation	278
5.72.3 Presentation Options	280
5.72.4 Example	280
5.72.5 Mapping	280
5.73 Creation/Destruction Markers	281
5.73.1 Semantics	281
5.73.2 Notation	281
5.73.3 Presentation options	281
5.73.4 Example	281
5.73.5 Mapping	282

Part 9 - Statechart Diagrams

5.74 Statechart Diagram	282
5.74.1 Semantics	282
5.74.2 Notation	282
5.74.3 Mapping	283
5.75 State	283
5.75.1 Semantics	283
5.75.2 Notation	283
5.75.3 Mapping	285
5.76 Composite States	285
5.76.1 Semantics	285
5.76.2 Notation	285
5.76.3 Examples	286
5.76.4 Mapping	287
5.77 Events	287
5.77.1 Semantics	287
5.77.2 Notation	288
5.77.3 Example	289
5.77.4 Mapping	289
5.78 Simple Transitions	289
5.78.1 Semantics	289
5.78.2 Notation	290
5.78.3 Example	290
5.78.4 Mapping	290

5.79	Transitions to and from Concurrent States	291
5.79.1	Semantics	291
5.79.2	Notation	291
5.79.3	Example	291
5.79.4	Mapping	291
5.80	Transitions to and from Composite States	291
5.80.1	Semantics	291
5.80.2	Notation	292
5.80.3	Presentation Options	292
5.80.4	Example	292
5.80.5	Mapping	293
5.81	Factored Transition Paths	294
5.81.1	Semantics	294
5.81.2	Notation	294
5.81.3	Examples	294
5.82	Submachine States	295
5.82.1	Semantics	295
5.82.2	Notation	296
5.82.3	Example	296
5.82.4	Mapping	297
5.83	Synch States	297
5.83.1	Semantics	297
5.83.2	Notation	297
5.83.3	Example	297
5.83.4	Mapping	297

Part 10 - Activity Diagrams

5.84	Activity Diagram	298
5.84.1	Semantics	298
5.84.2	Notation	298
5.84.3	Example	299
5.84.4	Mapping	300
5.85	Action State	300
5.85.1	Semantics	300
5.85.2	Notation	300
5.85.3	Presentation options	300
5.85.4	Example	300
5.85.5	Mapping	300
5.86	Subactivity state	300
5.86.1	Semantics	300
5.86.2	Notation	301
5.86.3	Example	301
5.86.4	Mapping	301
5.87	Decisions	301
5.87.1	Semantics	301
5.87.2	Notation	301
5.87.3	Example	302
5.87.4	Mapping	302
5.88	Call States	302

5.88.1 Semantics	302
5.88.2 Notation	302
5.88.3 Example	302
5.88.4 Mapping	303
5.89 Swimlanes	303
5.89.1 Semantics	303
5.89.2 Notation	303
5.89.3 Example	304
5.89.4 Mapping	304
5.90 Action-Object Flow Relationships	304
5.90.1 Semantics	304
5.90.2 Notation	305
5.90.3 Example	306
5.90.4 Mapping	306
5.91 Control Icons	306
5.91.1 Notation	307
5.91.2 Mapping	308
5.92 Synch States	308
5.93 Dynamic Invocation	309
5.93.1 Semantics	309
5.93.2 Notation	309
5.93.3 Mapping	309
5.94 Conditional Forks	309

Part 11 - Implementation Diagrams

5.95 Component Diagram	310
5.95.1 Semantics	310
5.95.2 Notation	310
5.95.3 Example	311
5.95.4 Mapping	312
5.96 Deployment Diagram	312
5.96.1 Semantics	312
5.96.2 Notation	312
5.96.3 Example	313
5.96.4 Mapping	313
5.97 Node	313
5.97.1 Semantics	313
5.97.2 Notation	314
5.97.3 Example	314
5.97.4 Mapping	315
5.98 Component.....	315
5.98.1 Semantics	315
5.98.2 Notation	316
5.98.3 Example	316
5.98.4 Mapping	317

6 UML Example Profiles	319
------------------------------	-----

Example 1 - UML Profile for Software Development Processes

6.1	Introduction	319
6.2	Summary of Profile.....	319
6.3	Stereotypes and Notation	320
6.3.1	Use Case Stereotypes	320
6.3.2	Analysis Stereotypes	321
6.3.3	Design Stereotypes.....	322
6.3.4	Implementation Stereotypes	323
6.3.5	Class Stereotypes	324
6.3.6	Association Stereotypes	325
6.4	Well-Formedness Rules	325
6.4.1	Generalization	326
6.4.2	Containment.....	326
Example 2 - UML Profile for Business Modeling		
6.5	Introduction	326
6.6	Summary of Profile.....	326
6.7	Stereotypes and Notation	327
6.7.1	Use Case Stereotypes	327
6.7.2	Organization Stereotypes	328
6.7.3	Class Stereotypes	329
6.7.4	Association Stereotypes.....	331
6.8	Well-Formedness Rules	332
6.8.1	Generalization	332
7	UML Model Interchange	333
7.1	Overview	333
7.2	Model Interchange Using XMI	353
7.3	Model Interchange Using CORBA IDL	355
8	Object Constraint Language Specification	357
8.1	Overview	357
8.1.1	Why OCL?.....	357
8.1.2	Where to Use OCL.....	357
8.2	Introduction	358
8.2.1	Legend	358
8.2.2	Example Class Diagram.....	358
8.3	Relation to the UML Metamodel.....	359
8.3.1	Self	359
8.3.2	Specifying the UML context	359
8.3.3	Invariants	360
8.3.4	Pre- and Postconditions	360
8.3.5	Package context	361
8.3.6	General Expressions	361
8.4	Basic Values and Types	361
8.4.1	Types from the UML Model.....	362
8.4.2	Enumeration Types.....	362
8.4.3	Let Expressions and «definition» Constraints	362
8.4.4	Type Conformance.....	363
8.4.5	Re-typing or Casting	364

8.4.6	Precedence Rules	364
8.4.7	Use of Infix Operators	364
8.4.8	Keywords	365
8.4.9	Comment	365
8.4.10	Undefined Values	365
8.5	Objects and Properties	366
8.5.1	Properties	366
8.5.2	Properties: Attributes.....	366
8.5.3	Properties: Operations	366
8.5.4	Properties: Association Ends and Navigation	367
8.5.5	Navigation to Association Classes	368
8.5.6	Navigation from Association Classes	369
8.5.7	Navigation through Qualified Associations	370
8.5.8	Using Pathnames for Packages.....	370
8.5.9	Accessing overridden properties of supertypes	370
8.5.10	Predefined properties on All Objects	371
8.5.11	Features on Classes Themselves	372
8.5.12	Collections.....	373
8.5.13	Collections of Collections	374
8.5.14	Collection Type Hierarchy and Type Conformance Rules	374
8.5.15	Previous Values in Postconditions	374
8.6	Collection Operations	375
8.6.1	Select and Reject Operations	375
8.6.2	Collect Operation	377
8.6.3	ForAll Operation	378
8.6.4	Exists Operation	378
8.6.5	Iterate Operation	379
8.6.6	Iterators in Collection Operations	380
8.6.7	Resolving Properties	380
8.7	The Standard OCL Package	380
8.8	Predefined OCL Types	381
8.8.1	Basic Types	381
8.8.2	Collection-Related Types	388
8.9	Grammar	397
A	UML Standard Elements	403
B	Legal Information	407
	Glossary	411
	Index	423

Preface

The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components.

The UML represents the culmination of best practices in practical object-oriented modeling. The UML is the product of several years of hard work, in which we focused on bringing about a unification of the methods most used around the world, the adoption of good ideas from many quarters of the industry, and, above all, a concentrated effort to make things simple.

We mean "we" in the most general sense. The three of us started the UML effort at Rational and were its original chief methodologists, but the final product was a team effort among many UML partners under the sponsorship of OMG. All partners came with their own perspectives, areas of concern, and areas of interest; this diversity of experience and viewpoints has enriched and strengthened the final result. We extend our personal thanks to everyone who was a part of making the UML a reality. We would like to thank Rational for giving us the opportunity to work freely so that we might focus on unification, and we want to recognize all the other companies representing the UML partners for seeing the importance of the UML to the industry as a whole and giving their representatives time to work on this project. We must also thank the OMG for providing the framework under which we could bring together many diverse opinions to develop a consensus result. We expect that OMG's ownership of the UML standard and the public's free access to it will ensure the widespread use and advancement of UML technology over the coming years.

In an effort that involved so many companies and individuals with so many agendas, one would think that the resulting product would be the software equivalent of a camel: a most dysfunctional-looking animal that appears to have been the work product of an ill-formed committee of misfits. The UML most decidedly is not a random collection of political compromises. If anything, because of the focus we placed upon creating a complete and formal model, the UML is coherent and has harmony of design.

In this context it is also exciting to point out that the UML was developed alongside, and with the full collaboration, of the OMG's Meta-Object Facility (MOF) team. The MOF, which represents the state of the art in distributed object repository architectures, is OMG's adopted technology for modeling and representing metadata (including the UML metamodel) as CORBA objects. The UML and MOF standards are key building blocks of OMG's development environment for building and deploying distributed object systems.

It is a very real sign of maturity of the industry that the UML exists as a standard. At a time when software is increasingly more complex and more central to the mission of companies and countries, the UML comes at the right time to help organizations deal with this complexity. Already, without a lot of the fanfare or hype sometimes associated with programming languages, the UML is in use in hundreds (if not thousands) of projects around the world, a sign that it is part of the mainstream of engineering software.

Grady Booch
Ivar Jacobson
Jim Rumbaugh
Rational Software Corporation

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19501 was prepared by Technical Committee ISO/IEC/TC JTC1, Information technology, Subcommittee SC 7, Software and System Engineering in collaboration with the Object Management Group (OMG), following the submission and processing as a Publicly Available Specification (PAS) of the OMG Unified Modeling Language (UML) specification Version 1.4.2.

While not limited to this context, the UML standard is closely related to work on the standardization of Open Distributed Processing (ODP), the coordinating framework for which is provided by ITU-T Recommendations X.901-904 | ISO/IEC 10746, the Reference Model of Open Distributed Processing (RM-ODP).

Apart from this Foreword, the text of this International Standard is identical with that for the OMG specification for UML 1.4.2 (OMG reference formal/04-07-02).

Introduction

The Unified Modeling Language (UML) is a general-purpose modeling language with a semantic specification, a graphical notation, an interchange format, and a repository query interface. It is designed for use in object-oriented software applications, including those based on technologies recommended by the Object Management Group (OMG). As such, it serves a variety of purposes including, but not limited to, the following:

- a means for communicating requirements and design intent,
- a basis for implementation (including automated code generation),
- a reverse engineering and documentation facility.

As an international standard, the various components of UML provide a common foundation for model and metadata interchange:

- between software development tools,
- between software developers, and
- between repositories and other object management facilities.

The existence of such a standard facilitates the communication between standardized UML environments and other environments.

While not limited to this context, the UML standard is closely related to work on the standardization of Open Distributed Processing (ODP).

The rapid growth of distributed processing has led to a need for a coordinating framework for this standardization and ITU-T Recommendations X.901-904 | ISO/IEC 10746, the Reference Model of Open Distributed Processing (RM-ODP) provides such a framework. It defines an architecture within which support of distribution, interoperability and portability can be integrated.

RM-ODP Part 2 (ISO/IEC 10746-2) defines the foundational concepts and modeling framework for describing distributed systems. The scopes and objectives of the RM-ODP Part 2 and the UML, while related, are not the same and, in a number of cases, the RM-ODP Part 2 and the UML specification use the same term for concepts which are related but not identical (e.g., interface). Nevertheless, a specification using the Part 2 modeling concepts can be expressed using UML with appropriate extensions (using stereotypes, tags and constraints).

RM-ODP Part 3 (ISO/IEC 10746-3) specifies a generic architecture of open distributed systems, expressed using the foundational concepts and framework defined in Part 2. Given the relation between UML as a modeling language and Part 2 of the RM ODP standard, it is easy to show that UML is suitable as a notation for the individual viewpoint specifications defined by the RM-ODP.

Structure of this standard

Chapters 1-3: Scope, Normative References, and General Information.

Chapter 4: UML Semantics - Specifies semantics for structural and behavioral object models. Structural models (also known as static models) emphasize the structure of objects in a system, including their classes, interfaces, attributes and relations.

Chapter 5: UML Notation Guide - Describes the notation for the visual representation of the Unified Modeling Language (UML). This notation document contains brief summaries of the semantics of UML constructs, but the UML Semantics chapter must be consulted for full details.

Chapter 6: UML Example Profiles - Contains these examples: Example 1: UML Profile for Software Development Processes and Example 2 - UML Profile for Business Modeling.

Chapter 7: UML Model Interchange - UML model interchange is based on the Metaobject Facility (MOF) 1.3 Specification. The UML Semantics abstract syntax is mapped to a set of MOF packages called the UML Interchange Metamodel.

Chapter 8: Object Constraint Language Specification - Introduces and defines the Object Constraint Language (OCL), a formal language used to express constraints.

Annex A: UML Standard Elements

Annex B: Standard Legal Information

Information Technology - Open Distributed Processing - Unified Modeling Language (UML) Version 1.4.2

1 Scope

This standard specifies the Unified Modeling Language (UML) with the objective of providing system architects working on object analysis and design with one consistent language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling.

This standard represents the convergence of best practices in the object-technology industry. UML is the proper successor to the object modeling languages of three previously leading object-oriented methods (Booch, OMT, and OOSE). The UML is the union of these modeling languages and more, since it includes additional expressiveness to handle modeling problems that these methods did not fully address.

One of the primary goals of UML is to advance the state of the industry by enabling object visual modeling tool interoperability. However, in order to enable meaningful exchange of model information between tools, agreement on semantics and notation is required. UML meets the following requirements:

- Formal definition of a common object analysis and design (OA&D) metamodel to represent the semantics of OA&D models, which include static models, behavioral models, usage models, and architectural models.
- IDL specifications for mechanisms for model interchange between OA&D tools. This document includes a set of IDL interfaces that support dynamic construction and traversal of a user model.
- A human-readable notation for representing OA&D models. This document defines the UML notation, an elegant graphic syntax for consistently expressing the UML's rich semantics. Notation is an essential part of OA&D modeling and the UML.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid.

All Recommendations and Standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1995, OpenDistributed Processing - Reference Model: Foundations
- ITU-T Recommendation X.903 (1995) | ISO/IEC 10746-3:1995, OpenDistributed Processing - Reference Model: Architecture
- ISO/IEC 15474-1:2002(E): Information technology - CDIF framework - Part 1: Overview

ISO/IEC 19501::2005(E)

- ISO/IEC 15474-2:2002(E): Information technology - CDIF framework - Part 2: Modelling and extensibility
- ISO/IEC 15475-1:2002(E): Information technology - CDIF transfer format - Part 1: General rules for syntaxes and encodings
- ISO/IEC 15475-2:2002(E): Information technology - CDIF transfer format - Part 2: Syntax SYNTAX.1
- ISO/IEC 15475-3:2002(E): Information technology - CDIF transfer format - Part 3: Encoding ENCODING.1
- ISO/IEC 15476-1:2002(E): Information technology - CDIF semantic metamodel - Part 1: Foundation
- ISO/IEC 15476-2:2002(E): Information technology - CDIF semantic metamodel - Part 2: Common
- ISO/IEC 15476-3 (under development): Information technology - CDIF semantic metamodel - Part 3: Data Definition
- ISO/IEC 15476-4 (under development): Information technology - CDIF semantic metamodel - Part 4: Data Models
- ISO/IEC 15476-5 (under development): Information technology - CDIF semantic metamodel - Part 5: Data Flow Models
- ISO/IEC 15476-6 (under development): Information technology - CDIF semantic metamodel - Part 5: State/Event Models