

ÍNDICE

COORDINADORES.....	17
AUTORES	21
PRÓLOGO	37
PREFACIO.....	43
CAPÍTULO 1. DESARROLLO DIRIGIDO POR MODELOS: UN NUEVO PARADIGMA DE CONSTRUCCIÓN DE SOFTWARE	51
1.1 INTRODUCCIÓN.....	51
1.2 DE LOS OBJETOS A LOS MODELOS.....	52
1.3 PRINCIPIOS BÁSICOS DEL MDE.....	56
1.4 LOS PRINCIPALES PARADIGMAS MDE.....	60
PARTE I. METAMODELADO.....	65
CAPÍTULO 2. CONCEPTOS BÁSICOS DE MODELADO	67
2.1 INTRODUCCIÓN: QUÉ ES UN MODELO.....	67
2.2 SEMÁNTICA: EL SIGNIFICADO DE UN MODELO	70
2.3 SINTAXIS: LA ESTRUCTURA DE UN MODELO	74
2.4 RELACIÓN ENTRE SEMÁNTICA Y SINTAXIS.....	77
2.5 CONCLUSIÓN: TOMEMOS EN SERIO LOS DIAGRAMAS.....	79
2.6 PARA SABER MÁS.....	79
CAPÍTULO 3. UNA INTRODUCCIÓN AL METAMODELADO.....	81
3.1 EL CONCEPTO DE METAMODELO	82

3.2 LENGUAJES DE METAMODELADO.....	88
3.3 ARQUITECTURA DE CUATRO NIVELES	91
3.4 ARQUITECTURA DE METAMODELO	96
3.5 CONCLUSIONES	101
CAPÍTULO 4. EL LENGUAJE OCL.....	103
4.1 INTRODUCCIÓN	103
4.2 DEFINICIÓN DE RESTRICCIONES DE INTEGRIDAD	106
4.2.1 Identificación de la instancia contextual	106
4.2.2 Navegación entre clases de objetos.....	107
4.2.3 Selección de un subconjunto de objetos.....	109
4.2.4 Condición aplicable a todos los objetos de una colección	110
4.2.5 Navegación por jerarquías de especialización	111
4.2.6 Restricción de clave primaria de una clase de objetos.....	112
4.2.7 Obtención de todos los objetos de una clase.....	113
4.2.8 Resumen de operaciones aplicables a una colección	114
4.3 DEFINICIÓN DE INFORMACIÓN DERIVADA.....	115
4.4 DEFINICIÓN DE OPERACIONES	116
4.5 HERRAMIENTAS DE SOPORTE	121
4.6 CONCLUSIONES	122
CAPÍTULO 5. PERFILES MOF.....	123
5.1 INTRODUCCIÓN	123
5.2 PERFILES EN LA ARQUITECTURA MOF.....	124
5.3 MECANISMO DE PERFILES EN UML 2.0	125
5.3.1 Perfil.....	127
5.3.2 Estereotipos y propiedades.....	128
5.3.3 Restricciones	128
5.4 MÉTODO PARA LA DEFINICIÓN DE PERFILES	128
5.5 EJEMPLO	129
5.6 ¿METAMODELOS O PERFILES?.....	132
5.7 CONCLUSIONES	132
PARTE II. SINTAXIS CONCRETA	135
CAPÍTULO 6. ECLIPSE GRAPHICAL MODELING FRAMEWORK	137
6.1 INTRODUCCIÓN	137
6.1.1 ¿Qué es y para qué sirve GMF?	137
6.1.2 Relación de GMF con otros <i>plug-ins</i> de Eclipse.....	138

6.2	INSTALACIÓN Y CREACIÓN DE PROYECTOS GMF	139
6.2.1	Instalación de GMF	139
6.2.2	Creación y estructura de los proyectos GMF	139
6.3	CREACIÓN DE UN EDITOR GRÁFICO SENCILLO	142
6.3.1	Construcción del editor gráfico por defecto.....	143
6.3.2	Generación y ejecución del editor gráfico	147
6.3.3	Algunos trucos sencillos para mejorar nuestros editores	148
6.4	ASPECTOS AVANZADOS: CONTENCIÓN GRÁFICA	150
6.5	CONCLUSIONES	153
CAPÍTULO 7. EUGENIA		155
7.1	INTRODUCCIÓN	155
7.1.1	Epsilon.....	155
7.1.2	Instalación de Epsilon	157
7.2	¿QUÉ ES Y PARA QUÉ SIRVE EUGENIA?	157
7.3	RELACIÓN DE EUGENIA CON OTROS <i>PLUG-INS</i> DE ECLIPSE	157
7.4	DESARROLLO DE UN EDITOR GRÁFICO CON EUGENIA.....	158
7.4.1	Introduciendo anotaciones GMF en los metamodelos	159
7.4.2	Desarrollo de un editor básico.....	162
7.4.3	Mejorando los editores desarrollados con EuGENia	167
7.5	ALGUNOS ASPECTOS AVANZADOS	169
7.5.1	Generación del editor en un solo paso	169
7.5.2	Invocación automática mediante tareas ANT	170
7.6	CONCLUSIONES	171
CAPÍTULO 8. CREACIÓN DE EDITORES TEXTUALES DE MODELOS CON XTEXT		173
8.1	INTRODUCCIÓN	173
8.2	CREACIÓN DE UN EDITOR TEXTUAL SENCILLO	174
8.2.1	Definición del metamodelo	175
8.2.2	Creación del editor por defecto con Xtext	175
8.2.3	Ejecución del editor textual de modelos generado.....	178
8.3	IMPLEMENTACIÓN DE UN EDITOR AVANZADO.....	180
8.3.1	Extensión del lenguaje de modelado.....	180
8.3.2	Creación del editor por defecto con Xtext	181
8.3.3	Importación de ficheros.....	182
8.3.4	Mejora del proceso de validación de los modelos	183
8.3.5	Modificación del estilo del texto.....	184
8.3.6	Mejora del asistente de contenido	186

8.3.7 Serialización de los modelos	189
8.4 CONCLUSIONES	190
CAPÍTULO 9. MICROSOFT DSL TOOLS	191
9.1 INTRODUCCIÓN	192
9.1.1 Relación con Visual Studio	192
9.2 CREACIÓN DE UN EDITOR GRÁFICO SENCILLO CON DSL TOOLS	195
9.2.1 El dominio específico: personas y deudas	195
9.2.2 Detalles gráficos	197
9.3 UN EJEMPLO MÁS COMPLEJO	200
9.3.1 Algunos aspectos avanzados: personalización	203
9.4 ESTRUCTURA DE UN PROYECTO DE DSL TOOLS	205
9.4.1 Arquitectura de los lenguajes generados con DSL Tools	206
9.5 CONCLUSIONES	208
PARTE III. TRANSFORMACIONES	209
CAPÍTULO 10. TRANSFORMACIONES DE MODELOS	211
10.1 INTRODUCCIÓN	211
10.2 APROXIMACIONES PARA EL DESARROLLO DE TRANSFORMACIONES	212
10.3 TRANSFORMACIONES DE MODELOS	215
10.3.1 Lenguajes de transformación de modelo a modelo	217
10.4 GENERACIÓN DE CÓDIGO	224
10.4.1 Lenguajes de transformación modelo a texto	226
10.5 PARA SABER MÁS	228
CAPÍTULO 11. EL LENGUAJE ATL	231
11.1 INTRODUCCIÓN	231
11.2 UNIDADES PRINCIPALES DE ATL	232
11.2.1 Módulo ATL	232
11.2.2 <i>Query</i> ATL	233
11.2.3 Librerías ATL	233
11.3 EL LENGUAJE ATL	234
11.3.1 Encabezado	235
11.3.2 Importación de librerías	235
11.3.3 Helper Rules	236
11.3.4 Matched Rules	237
11.3.5 [Unique] Lazy Rules	239
11.3.6 Called Rules	240

11.3.7 Fases de ejecución de una transformación	241
11.3.8 Referenciando elementos creados en otras reglas	241
11.3.9 Código imperativo en ATL	243
11.3.10 Tipos de datos.....	244
11.3.11 Recomendaciones.....	247
CAPÍTULO 12. EL LENGUAJE QVT: QUERY/VIEW/TRANSFORMATION.....	249
12.1 INTRODUCCIÓN	249
12.2 COMPONENTES BÁSICOS	250
12.2.1 El lenguaje QVT Relations	250
12.2.2 El lenguaje QVT Core.....	251
12.2.3 El lenguaje QVT Operational Mappings.....	251
12.3 EL LENGUAJE QVT OPERATIONAL MAPPINGS	251
12.3.1 Declaración de una transformación.....	252
12.3.2 Operación <i>main</i>	254
12.3.3 Operación de <i>mapping</i>	255
12.3.4 Operaciones <i>helper</i> y <i>query</i>	260
12.3.5 Constructores.....	261
12.3.6 Bibliotecas.....	262
12.3.7 Reutilización de transformaciones	262
12.3.8 Operaciones y operadores usados en el cuerpo de las operaciones de <i>mapping</i> y de <i>query</i>	263
12.4 ENTORNO DE DESARROLLO	268
12.5 CONCLUSIONES	270
CAPÍTULO 13. EL LENGUAJE RUBYTL	271
13.1 INTRODUCCIÓN	271
13.2 EL LENGUAJE DE TRANSFORMACIÓN RUBYTL	272
13.2.1 Estructura del lenguaje	274
13.2.2 Semántica de ejecución	276
13.2.3 Reglas de nombrado	278
13.2.4 Manipulación de modelos	279
13.2.5 Modularización de transformaciones	284
13.3 ENTORNO DE DESARROLLO	285
13.3.1 Configuración de transformaciones	285
13.3.2 Mensajes de error	286
13.4 CONCLUSIONES	286

CAPÍTULO 14. EL LENGUAJE JET	289
14.1 CONCEPTOS BÁSICOS DE JET	289
14.1.1 Sintaxis JET.....	290
14.1.2 Uso de XPath en JET	295
14.2 TRANSFORMACIONES JET	297
14.2.1 Proyectos JET.....	297
14.2.2 Plantillas JET.....	298
14.2.3 Etiquetas JET.....	299
14.3 EJEMPLO M2T CON JET.....	303
CAPÍTULO 15. EL LENGUAJE MOFSCRIPT	307
15.1 REQUISITOS, INSTALACIÓN Y VISTA GENERAL.....	308
15.2 EJEMPLO: GENERACIÓN DE CÓDIGO SQL	311
15.3 EL LENGUAJE MOFSCRIPT	314
15.3.1 Importación de transformaciones previas: import.....	314
15.3.2 Definición de la transformación: texttransformation	314
15.3.3 Tipos de datos, variables y propiedades.....	315
15.3.4 Definición de reglas	317
15.3.5 Sentencias de selección: If y Select	318
15.3.6 Sentencias de repetición: While y ForEach	319
15.3.7 Operaciones OCL destacadas de MOFScript.....	319
15.3.8 Bloque de texto protegido contra escritura	320
15.4 EJEMPLO DE TRANSFORMACIÓN: MÁQUINAS DE ESTADOS	321
15.4.1 Metamodelo de máquinas de estados.....	321
15.4.2 Transformación MOFScript.....	322
15.5 CONCLUSIONES Y CONSEJOS.....	322
CAPÍTULO 16. EL LENGUAJE XPAND	325
16.1 INTRODUCCIÓN	325
16.2 CONCEPTOS BÁSICOS DE XPAND.....	326
16.3 TRANSFORMACIONES XPAND	328
16.3.1 Proyecto generador para Xpand	328
16.3.2 Definición de reglas de transformación	332
16.3.3 Ejemplo de transformación de modelo de datos a Java	332
16.4 REGIONES PROTEGIDAS EN XPAND	340
16.5 PROGRAMACIÓN ORIENTADA A ASPECTOS EN XPAND.....	341
16.5.1 Punto de unión y punto de corte.....	342
16.5.2 Definición del nombre.....	342

16.5.3 Tipos de parámetros	343
16.5.4 Procedimiento.....	343
16.6 CONCLUSIONES	344
PARTE IV. APLICACIONES I. CONSTRUCCIÓN DE APLICACIONES SOFTWARE CON MDE.....	345
CAPÍTULO 17. GENERACIÓN DE PORTALES WEB	347
17.1 INTRODUCCIÓN	347
17.2 DESARROLLO DE UN PORTAL UTILIZANDO <i>PORTLETS</i>	349
17.3 EL METAMODELO <i>TAREAS</i>	350
17.4 EL METAMODELO <i>ORQUESTACIÓN</i>	351
17.5 EL METAMODELO <i>PRESENTACIÓN</i>	354
17.6 EL METAMODELO <i>EXO</i>	356
17.7 DE SOP A EXO: TRANSFORMACIONES.....	359
17.8 BENEFICIOS DEL DSDM	365
CAPÍTULO 18. RUX-TOOL: HERRAMIENTA PARA EL MODELADO DE INTERFACES DE USUARIO WEB 2.0.....	369
18.1 INTRODUCCIÓN	369
18.2 RUX-METHOD.....	370
18.2.1 IU abstracta, concreta y final	372
18.3 RUX-TOOL Y WEBRATIO: MODELADO RIA	373
18.3.1 Modelado de datos con WebRatio	374
18.3.2 Modelado de hipertexto con WebRatio.....	374
18.3.3 Modelado de presentación con RUX-Tool	376
18.4 RUX-TOOL Y WEBRATIO: CASO DE ESTUDIO.....	378
18.4.1 Descripción del caso práctico.....	378
18.5 CONCLUSIONES	389
CAPÍTULO 19. GENERACIÓN DE CÓDIGO ANSI-C DE MODELOS DE COMPONENTES UML PARA SISTEMAS EMBEBIDOS	391
19.1-ESTRUCTURA GLOBAL DSDM UTILIZADA PARA LA GENERACIÓN DE CÓDIGO	394
19.2 HERRAMIENTAS	395
19.3 ESTRUCTURA DEL PROYECTO.....	396
19.3.1 Proyecto FromUML2SimpleC	397
19.3.2 Proyecto MMSimpleC	398
19.3.3 Proyecto SimpleC2Code	399
19.4 DESCRIPCIÓN DEL CASO DE ESTUDIO	401

19.4.1 Controlador de puertas automáticas deslizantes	401
19.4.2 Metodología de diseño	402
19.4.3 Diseño del control básico de un control de puertas automáticas	403
19.5 ANÁLISIS DE LA TRANSFORMACIÓN DE UML2+MARTE A SIMPLEC	412
19.6 CONCLUSIONES	421
19.7 AGRADECIMIENTOS	423
CAPÍTULO 20. DESARROLLO DIRIGIDO POR MODELOS EN LA PRÁCTICA. MOSKITT Y EL DESARROLLO DE APLICACIONES DE GESTIÓN.....	425
20.1 INTRODUCCIÓN	425
20.1.1 Herramientas	426
20.1.2 Arquitectura.....	427
20.1.3 Generación de código y prototipos	428
20.2 DESARROLLO DIRIGIDO POR MODELOS EN LA PRÁCTICA	429
20.2.1 Modelos y arquitectura de las aplicaciones web	429
20.2.2 Proceso de desarrollo	431
20.3 CASO DE ESTUDIO: FACTURACIÓN	432
20.3.1 Requisitos del caso de estudio.....	433
20.4 CREACIÓN DEL PROYECTO MOSKITT.....	433
20.5 ESPECIFICACIÓN DE LA LÓGICA DE NEGOCIO. CONSTRUCCIÓN DE DIAGRAMAS DE CLASES	433
20.6 ESPECIFICACIÓN DE LA INTERFAZ DE USUARIO: MODELOS SKETCHER Y UIM	434
20.6.1 Creación de un nuevo modelo Sketcher.....	435
20.6.2 Diseño de ventanas.....	436
20.7 GENERACIÓN DE CÓDIGO OPENXAVA	441
20.7.1 La transformación UML2JPA.....	442
20.7.2 La transformación UIM2OX.....	443
20.8 CONCLUSIONES	444
CAPÍTULO 21. DISEÑO Y DESARROLLO DE INTERFACES DE USUARIO.....	445
21.1 INTRODUCCIÓN	445
21.2 MODELADO DE INTERFACES DE USUARIO CON MOSKITT	447
21.2.1 MOSKitt UIM	449
21.2.2 MOSKitt Sketcher.....	453
21.3 TRANSFORMACIONES Y GENERACIÓN DE CÓDIGO.....	456
21.3.1 Transformaciones entre modelos de interfaz de usuario.....	457
21.3.2 Transformaciones modelo a texto	458

21.4 CASO DE ESTUDIO: FACTURACIÓN	460
21.4.1 Estrategia para desarrollar la interfaz.....	460
21.4.2 Configurar el proyecto MOSKitt: Facturación	460
21.4.3 Crear el boceto inicial con MOSKitt Sketcher.....	462
21.4.4 Completar la especificación con MOSKitt UIM.....	465
21.4.5 Generar/Prototipar la aplicación	466
21.5 CONCLUSIONES	468
PARTE V. APLICACIONES II. OTRAS APLICACIONES DE MDE	469
CAPÍTULO 22. MODELADO DE PROCESOS	471
22.1 INTRODUCCIÓN	471
22.2 INGENIERÍA DE PROCESOS O MÉTODOS	473
22.3 MODELADO DE PROCESOS SOFTWARE CON SPEM	476
22.4 MODELADO DE PROCESOS DE NEGOCIO CON BPMN	483
22.5 CONCLUSIONES	490
CAPÍTULO 23. MODERNIZACIÓN DEL SOFTWARE.....	491
23.1 INTRODUCCIÓN	492
23.2 DEFINICIONES Y CONCEPTOS	494
23.2.1 Sistemas de información heredados	494
23.2.2 Reingeniería	495
23.2.3 Modernización dirigida por la arquitectura (ADM).....	496
23.3 MODERNIZACIÓN DEL DEL SOFTWARE MEDIANTE ADM.....	498
23.3.1 El ecosistema KDM	501
23.4 EJEMPLO DE MODERNIZACIÓN DE SOFTWARE.....	502
23.4.1 Contexto del ejemplo de modernización.....	502
23.4.2 El proceso de modernización propuesto	503
23.4.3 Resultados obtenidos.....	511
23.5 CONCLUSIONES	512
CAPÍTULO 24. MODERNIZACIÓN DIRIGIDA POR LA ARQUITECTURA EN LA TRANSFORMACIÓN DE UN <i>CORE BANKING</i>.....	515
24.1 INTRODUCCIÓN	515
24.2 PROBLEMA.....	516
24.3 SOLUCIÓN	517
24.3.1 Metodología	519
24.4 MODELADO DEL SISTEMA AS-IS	520
24.5 DEFINICIÓN DE BRECHAS Y SISTEMAS TO-BE.....	522

24.6 RESOLUCIÓN DE BRECHAS Y TRANSFORMACIÓN DE CÓDIGO	525
24.7 CONCLUSIONES	527
CAPÍTULO 25. MODELOS EN TIEMPO DE EJECUCIÓN	529
25.1 INTRODUCCIÓN	530
25.2 CONCEPTOS BÁSICOS	531
25.2.1 ¿Qué son los modelos en tiempo de ejecución?	531
25.2.2 ¿Por qué modelos en tiempo de ejecución?	532
25.2.3 ¿Qué constituye un modelo en tiempo de ejecución?	533
25.2.4 ¿Cómo se representan los modelos en tiempo de ejecución?	534
25.2.5 ¿Qué beneficios ofrecen los modelos en tiempo de ejecución en los sistemas autoadaptables?	535
25.3 APLICACIONES	536
25.3.1 Hogares inteligentes	537
25.3.2 Adaptación de interfaces de usuario en dispositivos móviles	539
25.4 CASO DE ESTUDIO	542
25.4.1 Definición del caso	542
25.4.2 Autoadaptación del sistema	544
25.4.3 El rol de los modelos en tiempo de ejecución en el ciclo de desarrollo de software	544
25.4.4 Tecnologías para gestionar modelos en tiempo de ejecución	546
25.5 CONCLUSIONES	551
BIBLIOGRAFÍA	553
MATERIAL ADICIONAL	575
ÍNDICE ALFABÉTICO	577