# La Representación de Recursos usando la metodología del Desarrollo Dirigido por Modelos en un Repositorio Institucional. Caso de estudio: SEDICI


**Autor:** Mg. Jose Texier

**Director:** Prof. Ing. Marisa Raquel De Giusti

**Co-Directora:** Dra. Silvia Gordillo

Facultad de Informática - Universidad Nacional de La Plata

Tesis presentada para obtener el grado de Doctor en Ciencias Informáticas

La Plata, República Argentina

Julio 2015

## Dedicatoria

A mi Papá, que siempre me enseñó que la constancia y la honestidad son los pilares para todo lo que uno realiza en la vida.

A mi Mamá, que desde que tengo uso de razón, solamente me pedía que estudiase ya que ella nunca tuvo las oportunidades que tuve.

A mi Jus, que siempre me ha apoyado y me dio las fuerzas necesarias en el momento que más lo necesitaba para salir adelante y animarnos a esta labor tan bonita (hacer el doctorado en el extranjero) en la mejor etapa de la vida.

A mi hijo Augusto, espero que este trabajo sea uno de tantos ejemplos que pueda transmitirle para que seas un hombre de bien y siempre haga las cosas de la mejor manera posible!!!

## Agradecimientos

En primer lugar a DIOS!!!

A mi directora Prof. Ing. Marisa, por la insistencia y el apoyo que me dio durante el proceso de formación en el doctorado.

A mi codirectora Dra. Silvia, por su orientación y los consejos para definir una propuesta acorde con el grado académico exigido.

A Jus por todo el apoyo, las sugerencias, comentarios y orientaciones que me dio a lo largo de estos estudios.

A mi Papá que siempre me ayudó, en especial, con todos los trámites engorrosos y burocráticos que exige el Estado Venezolano a quienes queremos estudiar fuera de nuestras fronteras.

A mi Mamá, a mi Hermana, a Carol, a Alex y a la Sra. Nelly, que siempre de manera directa o indirecta me han ayudado y dado el empuje necesario para salir adelante, sobre todo, en los momentos difíciles!!!

A Gonzalo Villarreal quien leyó esta Tesis con atención y me hizo comentarios invaluables.

A Ariel Lira quien a lo largo de mi formación siempre tuvo la mejor disposición para cualquier consejo y además formó parte de los lectores de esta tesis.

A Nestor Oviedo, quien intervino desde la propuesta inicial del doctorado hasta su formalización!

A todo el personal del PREBI-SEDICI por su colaboración y hacerme sentir uno más del grupo de trabajo.

A la UNET, mi segunda casa y mi alma mater!!!

A Willy y Ramiro, quienes en el 2009 en Cúcuta, me abrieron una ventana a la posibilidad de hacer el doctorado sobre Bibliotecas Digitales con la Profesora Marisa de Giusti.

Al personal del Posgrado de la Facultad de Informática, por su colaboración, atención y predisposición.

A todos mis amigos, aquí en la Argentina y los que están en Venezuela, quienes de una u otra forma han contribuido a terminar esta etapa académica.

# La Representación de Recursos bajo el Model-Driven en un Repositorio Institucional. Caso de estudio: SEDICI

## Resumen

En el 2003, nace el Servicio de Difusión de la Creación Intelectual (SEDICI) como el Repositorio Institucional (RI) de la Universidad Nacional de La Plata (UNLP), soportado por un desarrollo de software propio llamado Celsius DL. A finales del 2011 se realizó un estudio para migrar a otro software más actualizado y seleccionaron DSpace, plataforma de software de RI más usada en el mundo. El proceso de migración de Celsius DL a DSpace finalizó en el 2012. DSpace permitió contar con nuevas funcionalidades, logró una flexibilidad en los cambios estratégicos por parte de la gerencia del repositorio, permitió modificaciones en la imagen institucional sin afectar la lógica del repositorio y amplió la gestión de las tipologías de los recursos aceptados. DSpace, al igual que otras plataformas, presenta limitaciones en relación con la representación de los recursos –que consiste en el proceso de registrar en forma persistente un conjunto de datos como síntesis y reemplazo del objeto "real" para poder identificarlo, recuperarlo y distribuirlo–, tal situación fue uno de los principales problemas presentados en la transición de Celsius DL a DSpace.

Esta investigación se centró en describir el problema de la representación de recursos dentro del repositorio institucional SEDICI y en exponer una posible solución, todo bajo el enfoque Model-Driven. El objeto de estudio –La Representación de Recursos– es un problema recurrente que ha sido estudiado por algunos autores en otros ámbitos geográficos: Malizia (2010), Paganelli (2005), Gonçalves (2004), Fox (2004), Candela (2007), entre otros. No obstante, los trabajos revisados abordan el tema en forma general, no se toma en cuenta el recurso como el eje central.

Se planteó el **desarrollo de un marco de referencia que permita el desarrollo de funcionalidades a partir del diseño de un modelo flexible de acuerdo con la representación de recursos**, bajo el contexto de SEDICI, pero replicable a otros repositorios. Dicho planteamiento se relaciona con los elementos de un repositorio: recursos, esquemas de metadatos, almacenamiento y catalogación, y con los siguientes procesos funcionales de acuerdo con el material depositado y la norma ISO 14721: carga (ingest), almacenamiento (storage), catalogación (cataloging), indización (indexing), búsqueda (search engine) y navegación (browsing). Para lograrlo, se hizo un relevamiento bibliográfico del tema, se selecciono el paradigma de construcción de software Model-Driven para la implementación de la solución y se analizaron diferentes modelos conceptuales y modelos de datos para construir un marco de referencia que soporte el modelo propuesto y que esté relacionada con la norma ISO 14721.

El enfoque de Desarrollo de Software Dirigido por Modelos (en inglés Model-Driven Development, MDD) es un paradigma de construcción de software que asigna a los modelos un rol central con modelos que van desde los más abstractos a los más concretos. Este paradigma, además, proporciona un ambiente que permite a los interesados compartir sus puntos de vista y manipular las representaciones de las

entidades del dominio. Por ello en la propuesta de esta investigación, el marco de referencia se estructuró en 5 módulos o fases a partir del enfoque Model-Driven: DSL para el desarrollo del modelo flexible, transformación del modelo a un modelo relacional, transformación del modelo relacional a un script SQL para la creación de la base de datos, mapping de los recursos de Dspace-SEDICI a la base de datos creada, y, desarrollo de una aplicación en WebRatio para la visualización y exportación de los recursos.

Esta investigación dió respuesta al objetivo planteado y vinculó premisas devenidas de tres disciplinas: Ciencias de la Información, Ciencias Documentales y Ciencias de la Computación (LIS). Entretejer tales áreas en una propuesta relacionada con repositorios institucionales representó un aporte en un área de vacancia en la literatura

**Palabras Clave:** representación de recursos, repositorios institucionales, SEDICI, DSpace, Model-Driven.

# Índice de contenidos


Dedicatoria....1
Agradecimientos....2
Resumen....3
Índice de Figuras....8
Índice de Tablas....9
Introducción....10
CAPÍTULO 1: CONSIDERACIONES INICIALES....15
1.1. Planteamiento del problema....15
1.2. Justificación....17
1.3. Alcance y limitaciones....18
1.4. Objetivos....19
1.4.1. General....19
1.4.2. Específicos....19
1.5. Metodología de investigación....20
CAPÍTULO 2: REVISIÓN TEÓRICA....22
2.1. Bibliotecas Digitales (BD)....22
2.2. Repositorios Institucionales (RI)....26
2.2.1. Diferencias con las bibliotecas digitales....28
2.2.2. Funciones básicas de los RI....29
2.2.3. Módulos funcionales de los RI....31
2.2.3.1. Carga (Ingest)....31
2.2.3.2. Almacenamiento del archivo (Archival storage)....32
2.2.3.3. Gestión de datos (Data management)....32
2.2.3.4. Acceso (Access)....33
2.2.3.5. Planeamiento de la preservación (Preservation planning)....33
2.2.3.6. Administración del depósito (Administration)....34
2.3. Representación de Recursos (RR)....34
2.3.1. Los Recursos....36
2.3.2. Metadatos....37
2.3.3. Factores que inciden en la representación de recursos....41
2.4. Estándares a considerar en los RI....44
2.4.1. XML....44
2.4.2. OAI-PMH....45
2.4.3. SWORD....46
2.4.4. OpenSearch....46
2.4.5. RSS....46
2.4.6. XMI....47
2.4.7. XSD....47
2.4.8. MOF....47
2.4.9. Ecore....48
2.4.10. EMF....48
2.4.11. EMP....48
CAPÍTULO 3: EL MODEL-DRIVEN....49
3.1. ¿Qué es el Model-Driven?....50

3.2. Ciclo de vida del MDD....51
3.3. Transformaciones en MDA....52
3.4. Arquitectura de metamodelado de 4 capas en MDA....53
3.5. Modelado específico del dominio (DSM)....54
3.6. Enfoque MDE....54
3.7. Ingeniería web y MDWE....56
3.7.1. WebML....57
3.7.2. WebRatio....58
3.8. Marco de referencia....59
CAPÍTULO 4: CASOS DE ESTUDIO....61
4.1. Propuestas existentes bajo Model-Driven....61
4.2. WebRatio en los RI....62
4.3. Incorporación de recursos en SEDICI....65
4.3.1. Celsius DL....65
4.3.2. Portal de congresos y revistas....66
4.3.3. Scopus....67
CAPÍTULO 5: METODOLOGÍA PROPUESTA PARA LA REPRESENTACIÓN DE RECURSOS....69
5.1. Análisis del contexto....71
5.1.1. Elementos de la representación de recursos....71
5.1.1.1. Tipologías de recursos....71
5.1.1.2. Esquema de metadatos....72
5.1.1.3. Almacenamiento....77
5.1.1.4. Catalogación....77
5.1.2. Modelos para repositorios institucionales....81
5.1.2.1 Modelos conceptuales para repositorios institucionales....82
5.1.2.2 Modelos de datos para repositorios institucionales....86
5.1.2.3. Síntesis....90
5.1.3. La Norma ISO 14721....92
5.2. Diseño....95
5.2.1. DSL norma ISO 14721....95
5.2.2. Del modelo flexible al modelo relacional....97
5.2.3. Modelo relacional a script de creación....99
5.2.4. Incorporación de registros....100
5.2.5. Aplicaciones....101
5.3. Implementación....102
5.3.1. DSL norma ISO 14721....104
5.3.2. Del modelo flexible al modelo relacional....108
5.3.3. Modelo relacional a script de creación....110
5.3.4. Script de ingreso de registros....111
5.3.5. Aplicaciones....113
5.4. Resultados....116
5.4.1. DSL norma ISO 14721....116
5.4.2. Del modelo flexible al modelo relacional....117
5.4.3. Modelo relacional a script de creación....118
5.4.4. Script de ingreso de registros....118

5.4.5. Aplicaciones....118
5.5. Implicación....119
CAPÍTULO 6: CONSIDERACIONES FINALES....122
6.1. Conclusiones....122
6.2. Trabajos futuros....125
Referencias....128
Apéndices....137
Apéndice A....137
Código fuente – Fase 1....137
Código fuente – Fase 2....139
Código fuente – Fase 3....146
Código fuente – Fase 4....148
Código fuente – Fase 5....150
Apéndice B....156
Estadios de la investigación....156
Apéndice C....157
Índice de acrónimos....157

# Índice de figuras

Figura 1-1. Esquema simplificado de la investigación proyectiva.....20
Figura 1-2. Esquema simplificado de la investigación realizada.....21
Figura 2-1. Proveedores del Protocolo OAI-PMH.....45
Figura 3-1. Acrónimos del Model Driven Star.....50
Figura 3-2. Ciclo de vida de MDD.....52
Figura 3-3. Esquema de la Transformación.....52
Figura 3-4. Transformaciones en MDA.....53
Figura 3-5. Vista general de las relaciones entre los 4 niveles.....54
Figura 3-6. Visión general de MDE.....55
Figura 3-7. Arquitectura WebRatio.....59
Figura 3-8. Esquema general de la propuesta.....60
Figura 4-1. Proceso WebRatio.....64
Figura 5-1. Marco de referencia propuesto.....70
Figura 5-2. Modelo Funcional OAIS.....93
Figura 5-3. Information Package recomendado por la norma ISO 14721.....95
Figura 5-4. Esquema de transformación M2M.....98
Figura 5-5. Esquema de transformación M2T.....99
Figura 5-6. Metamodelo SimpleClass.ecore.....104
Figura 5-7. GMF o dashboard.....105
Figura 5-8. Domain Gen Model.....106
Figura 5-9. Graphical Def Model.....106
Figura 5-10. Tooling Def Model.....106
Figura 5-11. Mapping Model.....107
Figura 5-12. Diagram Editor Gen Model.....107
Figura 5-13. Editor del modelo Norma.XMI.....108
Figura 5-14. Esquema de transformación M2M.....108
Figura 5-15. Proceso de ejecución M2M – ATL.....109
Figura 5-16. Código de transformación M2M – ATL .....110
Figura 5-17. Esquema de transformación M2T.....110
Figura 5-18. Proceso de ejecución M2T – Acceleo.....111
Figura 5-19. Código de transformación M2T – Acceleo.....112
Figura 5-20. Código de transformación T2T – inserts .....113
Figura 5-21. Modelo navegacional – WebRatio .....114
Figura 5-22. Capa de presentación – WebRatio .....114
Figura 5-23. Modelo relacional – WebRatio.....115
Figura 5-24. Modelo obtenido del DSL.....117

# Índice de tablas

Tabla_5-1:Identificación-libro....80
Tabla_5-2:Caso_Uso-Fase1–DSL....96
Tabla_5-3:Caso_Uso-Fase2-M2M....98
Tabla_5-4:Caso_Uso-Fase3-M2T....100
Tabla_5-5:Caso_Uso-Fase4-T2T....101
Tabla_5-6:Caso_Uso-Fase5-WebRatio....102
Tabla_5-7:XSD_WebRatio....116
Table_A-1:Código_metamodelo_SimpleClass....137
Table_A-2:Código_ejemplo_modelo_SimpleClass....138
Table_A-3:Código_metamodelo_SimpleRDBMSecore....139
Table_A-4:Código_SimpleRDBMSgenmodel....139
Table_A-5:Código_ejemplo_normaXMI....140
Table_A-6:Código_ejemplo_basededatosXMI....142
Table_A-7:Código_transformaciónATL....142
Table_A-8:Código_SimpleRDBMSgenmodel....146
Table_A-9:Código_ejemplo_transformaciónM2T....147
Table_A-10:Código_ejemplo_salidadeM2T....147
Table_A-11:Código_transformaciónT2T....148
Table_A-12:Código_RR/Model....150
Table_A-13:Código_WebModel/Properties....150
Table_A-14:Código_WebModel/sv1/area1/page1....151
Table_A-15:Código_DataModel/Properties....152
Tabla_B-1:Niveles_Conocimiento_De_La_Investigación....156

# Introducción

El desarrollo de Internet y de las Tecnologías de Información y Comunicación (TIC) han revolucionado las dinámicas del mundo, hecho que se muestra en el cambio de época que se está viviendo, conocida como la "Era de la Información". Estas transformaciones no son una novedad para la sociedad, la última ocurrió hace aproximadamente más de 200 años cuando la Revolución Industrial condujo a la humanidad del agrarianismo al industrialismo. La génesis de este movimiento se puede observar en lo cultural, social, económico y tecnológico (Castells, 2009; De Souza-Silva, Cheaz Peláez, & Calderón Romero, 2001), lo que ha conducido a una sociedad de nativos e inmigrantes digitales (Prensky, 2001) que está reestructurando los procesos y la economía mundial, donde, definitivamente, la materia prima es la información. Por ello, la necesidad de la gestión, difusión y preservación de esa información a través de plataformas de software como las Bibliotecas Digitales (BD) y los Repositorios Institucionales (RI).

Las BD surgieron a partir de 1990 (Borgman, 1999; Crow, 2002; Lesk, 1997; Lynch, 2003; Texier, 2013; Van de Sompel, Payette, Erickson, Lagoze, & Warner, 2004) y se fueron consolidando en el mundo científico con el pasar de los años, hasta entrelazarse con el concepto y funcionalidad de los RI. En la última década el crecimiento en la cantidad de repositorios y su contenido puede verse en registros internacionales (OpenDOAR, 2014) y representan una fuente de información digital especializada (principalmente trabajos científicos y académicos), organizada y accesible para los usuarios de diversas áreas (Texier, 2013). Además, los RI están en consonancia con los ideales y objetivos del Acceso Abierto (Björk & Solomon, 2012; Suber, 2012), y contribuyen a repensar los procesos de publicación de artículos científicos (Piwowar, Day, & Fridsma, 2007).

La concepción tradicional que se tiene de bibliotecas es la de ofrecer documentos (libros, revistas, tesis, artículos, etc.) en soportes físicos a través de servicios de préstamos y consultas. Estas prácticas estaban limitadas por la falta de ejemplares o la existencia de algunos desactualizados. De manera, que junto al incremento de los recursos informáticos, Internet y el descenso de los costos para

adquirir recursos y servicios relacionados, se potenciaron en los últimos 20 años el diseño y la creación de las BD (Agenjo & Hernández, 2010), es decir, se inició la automatización de las bibliotecas tradicionales, con un crecimiento sostenido y en constante evolución. Esta área ha sido discutida y presentada en conferencias internacionales de las disciplinas de las Ciencias Documentales, de la Información y de la Computación (Chowdhury & Chowdhury, 1999; Nguyen & Chowdhury, 2011; Yang, Pomerantz, Wildemuth, & Fox, 2006) entre las cuales se pueden nombrar: Joint Conference on Digital Libraries (JCDL), Theory and Practice of Digital Libraries (TPDL) antes conocida como European Conference on Research and Advanced Technology for Digital Libraries (ECDL), International Conference on Asian Digital Libraries (ICADL) y el IEEE/TCDL (Technical Committee on Digital Libraries). En los trabajos de estas conferencias y en las publicaciones de revistas del área que figuran en SCImago Journal & Country Rank y que cuentan con 148 revistas (consultado en julio del 2014) en la categoría "*Library and Information Sciences*" (SJC, 2014), se evidencia la diversidad de conceptos y descripciones que abarcan las palabras "bibliotecas digitales" o "digital library".

Desde 1990 en adelante comienza a acuñarse este término de BD aunque los modelos conceptuales y de datos de las mismas recién surgen a partir de la siguiente década, principalmente, en proyectos de investigación financiados por entes gubernamentales derivados de las áreas de informática y bibliotecas, individuales o en colaboración (Liew, 2009; Yang et al., 2006). En los noventa, a través de seis proyectos de las Universidades de Michigan, Stanford, Berkeley, Santa Barbara, Illinois y Carnegie Mellon (Griffin, 1998); en Estados Unidos de América surge la Digital Library Initiative (DLI-1) integrada por National Science Foundation (NSF), Defense Advanced Research Projects Agency (DARPA) y National Aeronautics and Space Administration (NASA). El objetivo de DLI-1 fue desarrollar e implementar modelos de BD para recopilar, almacenar y hacer disponible la búsqueda, recuperación y procesamiento de documentos científicos y de investigación a través de las redes de comunicación. A partir de los logros obtenidos por DLI-1, se anunció un nuevo programa llamado Digital Libraries Initiative - Phase 2 (DLI-2), conformado por National Library of Medicine (NLM), Library of Congress (LC), Federal Bureau of Investigation (FBI) y National

Endowment for the Humanities (NEH), además de los organismos que llevaron a cabo la DLI-1 (DLI, 1998). Los resultados de estas dos iniciativas, en su mayoría, permitieron la consolidación y estudio de nuevos estándares aplicados hoy en día en las Bibliotecas Digitales.

De forma paralela a tales iniciativas de BD surgieron diferentes conceptos y descripciones de RI, los cuales han tenido su entrada en el ámbito científico desde principios de los años 2000 con los trabajos de Lynch, (2003) y Van de Sompel, Payette, Erickson, Lagoze, & Warner (2004). Los RI han cobrado importancia en la sociedad académica y científica porque representan una fuente de información digital especializada, organizada y accesible para los lectores de diversas áreas.

En cuanto al diseño y desarrollo de repositorios institucionales se requiere que las diferentes partes interesadas (desarrolladores, dueños del negocio y expertos del dominio) se pongan de acuerdo sobre un lenguaje neutral y de alto nivel para describir, discutir y negociar los servicios y la información que pueden ofrecer, motivo por el cual es deseable que la construcción de sistemas o aplicaciones para el dominio de los repositorios se realice bajo metodologías de construcción de software, con el fin de obtener productos de mayor calidad y reuso (Pons, Giandini, & Pérez, 2010). En particular, se destaca una metodología en crecimiento y soportada por tecnologías abiertas que es el Desarrollo de Software Dirigido por Modelos (Model-Driven Development Software – MDD) (Navarro, Cristóbal, Fernández-Chamizo, & Fernández-Valmayor, 2011).

Para julio de 2014, existían alrededor de 2650 repositorios institucionales (OpenDOAR, 2014), implementados y desarrollados con plataformas de software que tienen con base modelos conceptuales creados en los últimos 25 años, a saber: DELOS, FRBR, Norma ISO 14721, entre otros (Candela et al., 2007; Gonçalves, Fox, Watson, & Kipp, 2004). Por ello, es necesario desarrollar un marco de referencia que permita el desarrollo de funcionalidades en los RI a partir de la representación de recursos de un RI. Se destaca que un marco de referencia es un sistema de software que consiste en la unión del modelo de referencia del negocio y un patrón arquitectónico que pueda ser usado por los profesionales del software para crear y gestionar funcionalidades en el dominio de estudio (Bass, Clements, & Kazman, 2003; ISO, 2014).

La propuesta esta de tesis se centra en cómo presentar un marco de referencia que permita el desarrollo de funcionalidades en este dominio a través del Desarrollo de Software Dirigido por Modelos. Los RI se han convertido en sistemas de información complejos que están basados en tecnologías y características de diferentes áreas de conocimiento como bibliotecología, sistemas de información, recuperación de información, representación de información e interacción persona-computador (Guo, 2010). Ante lo cual el MDD es una metodología para la construcción de software que permite involucrar diversas tecnologías, facilita el reuso del software (unión de esfuerzos) y está dirigido por modelos (Pons et al., 2010).

Esta tesis se ha estructurado en los siguientes capítulos:

1. *Primer capítulo: consideraciones iniciales*

   Se describe y contextualiza el problema planteado, los objetivos de la propuesta, la justificación, los alcances y limitaciones, y una breve exposición de la metodología de la investigación, que guio el estudio.

2. *Segundo capítulo: revisión teórica*

   Se presenta una revisión bibliográfica, en la que se profundiza sobre el concepto de las bibliotecas digitales y repositorios institucionales, funciones básicas y módulos funcionales de los repositorios institucionales, y la representación de recursos.

3. *Tercer capítulo: metodología de desarrollo de software*

   Se expone todo el enfoque que se utilizó para el desarrollo de la propuesta de esta investigación, es decir, conceptos y estructura de la metodología de desarrollo de software dirigida por modelos, conocida como Model Driven.

4. *Cuarto capítulo: casos de estudio*

   Los casos de estudio que se describieron en este capítulo, muestran una orientación a la necesidad de desarrollar un marco de referencia que permita generar funcionalidades a partir de una representación de recursos general.

5. *Quinto capítulo: análisis, diseño, implementación y resultados de la propuesta*

   En este capítulo se observan cada una de las etapas de la implementación desarrollada para solucionar el problema expuesto en la investigación.

6. *Sexto capítulo: consideraciones finales*

Se presentan las conclusiones y trabajos futuros que surgieron del estudio. Dichas consideraciones sintetizan y abren puertas para líneas de investigación que den continuidad a lo ofrecido en esta investigación.

# CAPÍTULO 1
# CONSIDERACIONES INICIALES

## *1.1. Planteamiento del problema*

En el 2003 nace el Servicio de Difusión de la Creación Intelectual (SEDICI, 2013) como el repositorio institucional de la Universidad Nacional de La Plata (UNLP), con el objetivo de socializar el conocimiento generado en las diferentes áreas académicas de la Universidad. SEDICI, portal de acceso libre, estuvo soportado por un desarrollo de software propio en PHP, MySQL y Java, llamado Celsius DL, adaptado a estándares internacionales como XML, OAI-PMH, SOAP, etc., que permitía el depósito y búsqueda de recursos (objetos físicos o digitales), por ejemplo: artículos de publicaciones periódicas, preprints, tesinas de grado y tesis de postgrado, producciones multimediales, libros electrónicos, autores, revistas, eventos, comunidades, colecciones, entre otros. Celsius DL creció en funcionalidades por lo que el diseño, mantenimiento e implementación convirtieron a la plataforma de software de SEDICI en una herramienta compleja, principalmente, por dos razones: por una parte, porque se requería de mayor tiempo y recursos humanos y, por otra, porqué se hacía necesaria una actualización de tecnologías de manera que no estuviera en detrimento de nuevos desarrollos e investigaciones.

En razón de eso, para finales del 2011, se estudió la posibilidad de migrar a una plataforma que estuviera a la par de las nuevas tecnologías aplicadas al dominio y que fuera más amena para el usuario y para la gestión de recursos por parte del personal de SEDICI (De Giusti et al., 2011). Este análisis tenía el objetivo de comparar diferentes plataformas de software: DSpace, EPrints, FEDORA y Greenstone (Madalli, Barve, & Amin, 2012; Pyrounakis & Nikolaidou, 2009; Singh, Witt, & Dorothea, 2010; Tzoc,

2013), tomando en cuenta:

- licencias de uso libres y gratuitas,
- alto nivel de aceptación por parte de la comunidad de repositorios digitales,
- sección de administración,
- mecanismos de personalización,
- actualizaciones periódicas de las versiones de la plataforma,
- soporte técnico para administradores y desarrolladores,
- manual de usuario y técnico,
- diferentes esquemas de metadatos permitidos,
- performance para obtener tiempos de respuesta bajos,
- capacidades de escalabilidad, y,
- estándares que garanticen la interoperabilidad entre repositorios.

La conclusión del estudio determinó que DSpace (DSpace, 2014) era la herramienta de software que mejor se adaptaba a las necesidades de SEDICI (De Giusti et al., 2011). El proceso de migración de Celsius DL a DSpace finalizó en el 2012, con la implementación de una nueva plataforma que se denominó DSpace-SEDICI con nuevas funcionalidades como: búsquedas y faceting, proveedor de datos y de servicios de acuerdo con el protocolo OAI-PMH, autenticación de usuarios, facilidad en la indexación por parte de los buscadores web, etc., En otras palabras, se logró una flexibilidad en los cambios estratégicos por parte de la gerencia del repositorio, modificaciones en la imagen institucional sin afectar el modelo de negocio del repositorio (procesos internos de software) y ampliar la gestión de las tipologías de los recursos aceptados.

Celsius DL y DSpace-SEDICI son plataformas de repositorios cuyo principal proceso es registrar en forma persistente un conjunto de datos como síntesis y reemplazo del objeto "real" para poder identificarlo, recuperarlo y distribuirlo por parte de los usuarios. Este proceso, conocido como la representación de recursos (Texier, De Giusti, Oviedo, Lira, & Villarreal, 2013), por lo general, es llevado de diferentes formas en las mismas plataformas. Por ejemplo, en los directorios internacionales de RI se

observan diferentes personalizaciones en DSpace o representaciones diversas en otras plataformas usadas (OpenDOAR, 2014; ROAR, 2014).

Es importante destacar que DSpace, al igual que las demás plataformas, presenta limitaciones (procesos de depósitos, manejo de estadísticas y de las comunidades-colecciones-items, vocabularios controlados centrados en los autores, entre otras) en cuanto al proceso de la representación de los recursos en los repositorios (Kökörvcený & Bodnárová, 2010; Pyrounakis & Nikolaidou, 2009), situación particular que generó un inconveniente en la transición de Celsius DL a DSpace en SEDICI, ya que se migraron muchos recursos en forma separada e incluso se realizaron adaptaciones por incompatibilidad de ambos sistemas porque la representación de distintos recursos era variada.

La representación de recursos es un problema recurrente que ha sido estudiado en diversas formas por algunos autores Malizia, Bottoni, & Levialdi (2010), Paganelli & Pettenati (2006), Gonçalves et al. (2004), y, Candela et al. (2007). No obstante, estos trabajos abordan el tema mediante sistemas diseñados en forma general en los que no se toma en cuenta el recurso como el eje central.

Los recursos son todo objeto, físico o digital, que se puede describir a partir de un conjunto de datos específicos (metadatos) que lo distinguen entre otros objetos, siempre dentro del contexto de los repositorios institucionales. Por ejemplo: papers, reviews, tesis, libros, etc. Tal situación genera la necesidad de contar con una representación de recursos de forma general y flexible de manera indistinta a la plataforma que se use.

Por ello, el objetivo de esta investigación es proponer un marco de referencia, que permita definir un modelo general y flexible para representar los recursos de repositorios institucionales bajo una metodología de desarrollo de software que tome en cuenta las características necesarias y se adapte a las diversas tecnologías actuales y venideras, para lo cual se tomó como campo de prueba a SEDICI.

## *1.2. Justificación*

El desarrollar una tesis doctoral, implica entre otras cosas, poder identificar un área de vacancia para hacer un aporte al conocimiento, pilar fundamental para lograr la

suficiencia en investigación requerida para este grado académico. La unión de los dominios de los repositorios institucionales y del desarrollo de software dirigido por modelos determinan un área de vacancia. De tal manera que permitir un desarrollo en el mundo de los RI y MDD significa un aporte al estado del arte, con el propósito de que sirva de comienzo para el diseño e implementación de sistemas de software para repositorios y/o bibliotecas digitales, en el que todas las partes involucradas en el sistema (desarrolladores de software, gerentes del sistema y expertos del dominio) puedan hablar un mismo lenguaje y obtener aplicaciones de calidad. Asimismo, esta tesis doctoral se justifica sobre tres ejes:

- Desde lo teórico, pues el modelo es una abstracción teórica del mundo real y representa gráficamente un sistema dado, por ello, la propuesta se centra en el desarrollo de un modelo general y flexible a través de un marco de referencia que represente los recursos de un sistema de repositorios de forma independiente.
- Desde lo metodológico se propone bajo la metodología de desarrollo de software dirigida por modelos (Model-Driven) que permita el aporte de las partes involucradas y que los cambios deseados no influyan en todo el proceso de construcción del software.
- Desde lo práctico la implementación de esta propuesta está dirigida, en primera instancia, a solventar una situación en la representación de recursos en SEDICI, con la intención de que pueda aplicarse sobre otros repositorios con base en DSpace y otras plataformas de software como EPrints o FEDORA.

## *1.3. Alcance y limitaciones*

Esta investigación se centró en una solución para la plataforma de software de repositorios más usado actualmente, DSpace, en el repositorio institucional de la Universidad Nacional de La Plata, SEDICI. Los recursos que se tomaron en cuenta de SEDICI para la propuesta fueron: artículos, tesis (grado, especializaciones, maestrías y doctorados), libros, autores, instituciones y revistas, catalogados con años de creación comprendidos entre 1794-2013; y no se gestiona el documento digital (.PDF, .Doc, etc.) sino su ruta absoluta y relativa. La razón por la cual se seleccionaron estos seis recursos

responde a que representan ampliamente lo que es un repositorio institucional y extender la investigación sobre los 30 recursos de SEDICI, convertiría la investigación en un trabajo muy extenso e inviable para un trabajo unipersonal.

La propuesta siguió los principios de la metodología de desarrollo de software dirigido por modelos, paradigma de construcción de software, puesto que es independiente de los productores de software en cuanto a las estandarizaciones y portabilidad de los sistemas de software. De manera tal, que esta metodología asigna a los modelos un rol central y activo bajo el cual se derivan modelos que van desde los más abstractos a los concretos mediante transformaciones sucesivas e iteraciones.

Asimismo, el desarrollo de la propuesta se realizó con software de licencias abiertas, de manera de dar continuidad a la filosofía de acceso abierto que es la que sustenta la creación de los RI, por ello, con esta propuesta se obtiene un software bajo los mismos lineamientos y sin problemas de licenciamiento ni pagos.

Algunas limitaciones estuvieron relacionadas conque la bibliografía daba cuenta de soluciones como un todo y no centradas en la representación de los recursos. Por otra parte, la información de los diversos recursos de SEDICI se encuentra en diferentes formatos de archivos, razón por la cual, no permitió reunirlos en un metamodelo general siguiendo el enfoque Model-Driven.

## *1.4. Objetivos*

### 1.4.1. General

Proponer un marco de referencia que permita definir un modelo general y flexible para obtener la representación de recursos del Repositorio Institucional de forma independiente de las tecnologías usadas.

### 1.4.2. Específicos

1. Describir la representación de los recursos del repositorio institucional que permita identificar las funciones básicas del repositorio.
2. Analizar las propuestas que se han desarrollado en cuanto a la representación de los recursos de los Repositorios Institucionales.
3. Identificar los factores que inciden en cada uno de los problemas de la

representación de recursos del repositorio institucional.

4. Caracterizar los esquemas de metadatos, tipologías de los recursos y entidades abstractas existentes que afectan la representación de los recursos.
5. Determinar los posibles escenarios relacionados con las diferentes reglas de catalogación que pueden afectar la representación de los recursos.
6. Diseñar un marco de referencia que soporte un modelo flexible de datos para la representación de los recursos dentro del Repositorio Institucional SEDICI de la UNLP, en concordancia con los aspectos encontrados a lo largo del estudio.
7. Implementar un prototipo para el desarrollo de funcionalidades del Repositorio Institucional SEDICI a partir de la representación de los recursos.

## *1.5. Metodología de investigación*

Las investigaciones se caracterizan por la búsqueda de un conocimiento nuevo a través de un proceso sistemático de indagación de los eventos de estudio del problema en cuestión, orientados al logro de un conocimiento (Hurtado, 2008). El propósito de esta tesis era definir un marco de referencia para obtener una representación de recursos que respondiera al problema. Por tanto, se optó por una metodología proyectiva (Hurtado, 2008) porque implica la creación y diseño de una propuesta para los Repositorios Institucionales previo a un proceso de investigación de la Representación de Recursos.

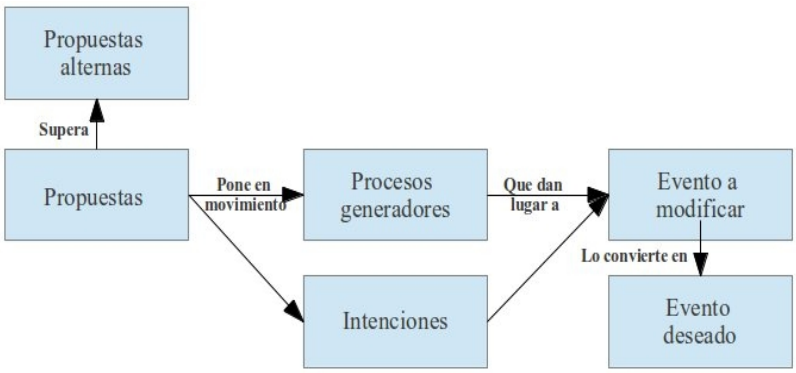


*Figura 1-1. Esquema simplificado de la investigación proyectiva. Fuente: Hurtado (2008)*

La investigación proyectiva remite a la creación, diseño o propuesta de algo con base en un proceso investigativo que debe proporcionar la información necesaria para desarrollarla, la cual amerita que una situación debe ser cambiada para mejorar un

proceso, identificando el evento a modificar (pueden ser varios) y, en algunos casos, el proceso generador (o causal) que da lugar a ese evento a modificar. Una versión del esquema de este tipo de investigación (Hurtado, 2008) se observa en la Figura 1-1.

A continuación se muestra el esquema adaptado a la investigación realizada, ver Figura 1-2:

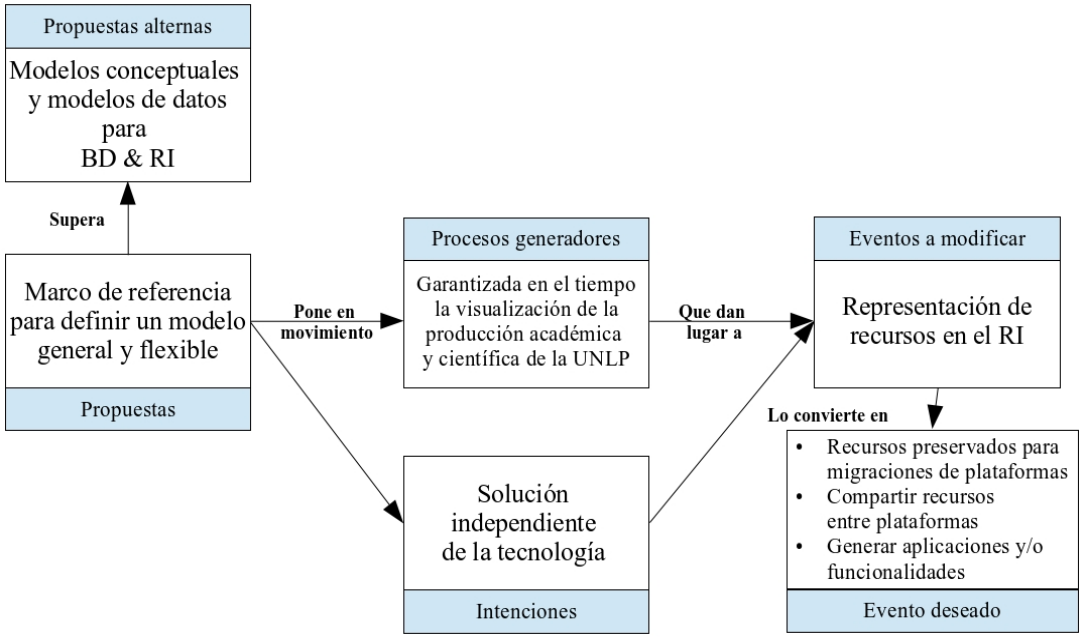


*Figura 1-2. Esquema simplificado de la investigación realizada*

Asimismo la investigación proyectiva, requiere pasar por una serie de objetivos específicos, que son de carácter integrativo, y que corresponden a los todos los estadios previos de conocimiento para llegar al objetivo general. Para complementar la formulación de estos objetivos, se puede ver en la Tabla B-1 en el Apéndice B.

# CAPÍTULO 2
# REVISIÓN TEÓRICA

La concepción tradicional de bibliotecas es la de ofrecer documentos en soportes físicos a través de servicios de préstamos y consultas. Ahora bien, este concepto ha evolucionado gracias al incremento de los recursos informáticos, Internet y el descenso de los costos para adquirir esos recursos y servicios, es decir, se inició la automatización de las bibliotecas tradicionales con el nacimiento de la Biblioteca Digital. A su vez surgieron los Repositorios Institucionales (RI) que tuvieron su entrada en el mundo científico desde principios del 2000. En este capítulo, el objetivo es exponer que el uso de los términos BD y RI, dependen del ámbito donde se aplica y, por ende, de los recursos con los que se desea trabajar. Asimismo se mostrará un estado del arte sobre los diferentes trabajos relacionados con el funcionamiento de un RI, la representación de los recursos, y las recomendaciones de estándares en el dominio.

## *2.1. Bibliotecas Digitales (BD)*

El concepto inicial que se pudo rastrear condujo a 1997, cuando Lesk (1997) planteó que las bibliotecas digitales son colecciones organizadas de información digital donde combinan la estructura y concurrencia de la información. Luego Waters (1998) expuso una definición para que todas las instituciones en la Digital Library Federation compartieran un mismo concepto que pudiera ser sometido a revisiones. Para este autor las BD consistieron en organizaciones que proveen los recursos tecnológicos para seleccionar, estructurar, controlar el acceso, conservar la integridad y asegurar la persistencia, a través del tiempo, de colecciones de trabajos digitales que estén disponibles para usarse por una o varias comunidades definidas.

Un año más tarde, Borgman (1999) establece que las bibliotecas digitales son un conjunto de recursos electrónicos y de capacidades técnicas asociadas para crear, buscar

y utilizar la información, es decir, extienden y mejoran los sistemas de almacenamiento y manipulación de datos tradicionales. El autor señala dos visiones, una referida a las comunidades de investigadores (contenido recopilado y organizado en virtud de comunidades de usuarios) y, la otra definición, referida a los profesionales de las bibliotecas (como instituciones u organizaciones que proveen servicios de información en formas digitales).

Por su parte Chowdhury & Chowdhury (1999) exponen un panorama general sobre el área y ponen en evidencia la proliferación de trabajos de investigación en los diferentes procesos presentes en las bibliotecas digitales, así como también la convergencia de bibliotecarios, informáticos, ingenieros electrónicos, abogados, científicos cognitivos, etc. En ese momento las BD estaban, en opinión de los autores, en una fase experimental incipiente.

De igual forma, en 1999 varios autores exponen que las BD utilizan las expresiones biblioteca electrónica, virtual y digital (Bawden & Rowlands, 1999b), que terminan siendo sinónimos en el área de estudio. Estos autores definen que una biblioteca electrónica estaría formada por objetos físicos que necesitan de medios electrónicos para el acceso a la información contenida. A la biblioteca virtual como aquella que hace uso de la realidad virtual para mostrar una interfaz y emular un ambiente que sitúe al usuario dentro de una biblioteca tradicional. Por su parte, la biblioteca digital remite al conjunto de materiales y servicios almacenados, procesados y accedidos en un repositorio de colecciones y contenidos digitalizados, almacenados en diferentes formatos electrónicos, procesados y accedidos mediante la utilización de herramientas y redes de comunicaciones digitales.

A través del método Delphi, Kochtanek & Hein (1999) analizan la percepción de los investigadores y usuarios sobre varios aspectos de la biblioteca digital, incluyendo su conceptualización. Las principales conclusiones a las que arribaron fueron que las iniciativas existentes hasta el momento son, principalmente, de colaboración. La biblioteca digital tiene el potencial para transformar el acceso a los registros digitales de conocimiento, en la cual el rol principal de los bibliotecarios es dar una extensión más al mundo de las bibliotecas digitales.

Para el 2001 Arms (2001) propone que una biblioteca digital es una colección

gestionada de información, con servicios asociados, donde la información es almacenada en formato digital accesible en toda la red, cuya definición enfatiza los aspectos de la gestión de los contenidos. Ese mismo año, se presentó un trabajo en el IFLA Annual Conference en Jerusalem, donde los autores Sharon & Frank (2001) definieron a la biblioteca digital como una biblioteca computarizada en donde la mayor parte de la información es digital y definieron una clasificación en 3 categorías: *Stand-alone Digital Library*: que consiste en una biblioteca simple en donde los recursos son digitales, *Federated Digital Library* es una federación de varias Stand-alone Digital Libraries independientes en la red, organizadas en un tema en común, y, *Harvested Digital Library* es una biblioteca virtual que proporciona acceso a material diseminado en la red.

En el 2002, Tramullas (2002) afirmó que las BD llegaron a un nivel de madurez, tal, que el objeto de estudio estuvo lo suficientemente definido por sus propias funciones y herramientas así como por los componentes tecnológicos, y define a la BD como un sistema de tratamiento técnico con acceso y transferencia de información digital, estructurado alrededor del ciclo de vida de una colección de documentos digitales, que ofrecen servicios interactivos de valor añadido para el usuario final. De igual manera, señala que más allá de que existan las bibliotecas "híbridas", la biblioteca digital emerge independientemente, en parte por el impulso enorme de un cuerpo de investigadores de la mitad a finales de la década de los noventa y, por el apoyo de las agencias gubernamentales.

Witten, Bainbridge & Nichols (2003) con la primera edición del libro "How to Build a Digital Library", afirman que las BD son colecciones enfocadas de objetos digitales (texto, vídeo y audio), métodos de acceso y recuperación de datos, para la selección, organización y mantenimiento de la colección. Tradicionalmente, estos se dividen en búsqueda y consulta, aunque en realidad la distinción no es clara.

Nuevamente Tramullas (2007) propone que no puede considerarse a las BD como bases de datos ni una digitalización de textos en Internet, tampoco como un OPAC (Online Public Access Catalog) integrado con enlaces y/o una colección de documentos en PDF o en cualquier otro formato. Por ello, define tres tipos principales de expresiones bibliotecarias: la biblioteca automatizada que es aquella en la cual los

procesos clásicos de tratamiento y recuperación son desarrollados mediante la utilización de sistemas informáticos. La biblioteca digitalizada que es aquella en la cual se han desarrollado procesos de digitalización de fondos por diferentes razones, y, finalmente, la biblioteca digital como aquella que pone a disposición de los usuarios una colección digital, sobre la cual se construyen servicios de valor añadido.

También en el 2007 se comienzan a observar modelos conceptuales de bibliotecas digitales, como es el caso del manifiesto DELOS donde se establece un marco conceptual con tres niveles (Candela et al., 2007):

- Biblioteca digital es la organización que recopila, gestiona, preserva y ofrece contenidos digitales,
- Sistema de biblioteca digital que se refiere al sistema de software que proporciona la funcionalidad requerida por una biblioteca digital particular y,
- Sistema de gestión de biblioteca digital que se refiere a la plataforma: sistema operativo, bases de datos, interfaz de usuario.

Más adelante, Guo (2010) define a las BD como aquellas que se procesan con tecnología digital y permite el almacenamiento de diversos documentos, además de presentarse como un sistema de información distribuido de recursos de información de diferentes fuentes y ubicaciones geográficas. Las características más importantes de las bibliotecas digitales que este autor expone son la digitalización de la colección, operaciones computacionales, transmisiones a través de la red, apertura del resguardo de la información y el compartir los recursos.

Para el 2011, se realiza un trabajo sobre los tópicos y subtópicos de las BD que fortalecen el concepto de las bibliotecas digitales y todas sus subáreas, con la creación de un mapa conceptual que permite crear un plan de estudios de un curso, ideal para investigadores de las BD, profesores, estudiantes y profesionales del área (Nguyen & Chowdhury, 2011).

En síntesis, las definiciones anteriores muestran que el concepto de Bibliotecas Digitales ha evolucionado en todos esos años, sobre todo, en los diferentes servicios ofrecidos y se fortalecieron con el crecimiento de la tecnología, representada por la Internet, el software y/o el hardware. En el entorno de trabajo de las BD surgen

actividades de investigación y desarrollo para sustentar la implementación y avanzar con diferentes objetivos y bajo diferentes perspectivas e interpretaciones. En razón de eso, en esta investigación se define una biblioteca digital como un sistema de información que permite el acceso y transferencia de información digital, estructurada alrededor de colecciones de documentos digitales sobre los cuales se ofrecen servicios a los usuarios. Este concepto no es definitivo, sino que está sujeto a actualizaciones a partir de cambios gracias a los avances tecnológicos, a las necesidades establecidas por los usuarios y a su relación con los Repositorios Institucionales.

### *2.2. Repositorios Institucionales (RI)*

Los Repositorios Institucionales (RI) están constituidos por un conjunto de archivos digitales en representación de productos científicos y académicos que pueden ser accedidos por los usuarios (Texier, 2013). Específicamente, los Repositorios Institucionales se entienden como estructuras web interoperables que alojan recursos científicos, académicos y administrativos, tanto físicos como digitales, descriptos por medio de un conjunto de datos específicos conocidos como metadatos (Lynch, 2003; Tramullas & Garrido, 2006; Van de Sompel et al., 2004). Tienen como propósito recopilar, catalogar, gestionar, acceder, difundir y preservar. Las actividades de catalogación, acceso, gestión y difusión de los contenidos son las más consolidadas con el crecimiento de los repositorios, por el contrario, la recopilación de materiales y la preservación todavía se encuentran en sus primeros pasos (De Giusti et al., 2011; Texier, 2013). Los RI son vías de la comunicación científica pero no son canales de publicación.

Los RI pueden estar agrupados en directorios de repositorios, los más importantes de acuerdo con enlaces entrantes o *inlinks* según Majestic (2014) y Ahrefs (2014) son: Directory of Open Access Repositories (OpenDOAR, 2014) con 2652 repositorios registrados, Registry of Open Access Repositories (ROAR, 2014) con 3646 repositorios registrados y University of Illinois OAI-PMH Data Provider Registry (UIUC, 2014) con 3018 repositorios (datos al 20 de julio del 2014).

Uno de los primeros autores que introduce el concepto de RI es Crow (2002), que los define como colecciones digitales que capturan y preservan la producción

intelectual de las comunidades universitarias.

Luego, Lynch (2003) señala que un repositorio institucional universitario es un conjunto de servicios que ofrece la universidad a los miembros de su comunidad, donde, es esencial un compromiso organizativo para la administración de estos materiales digitales, que incluye la preservación a largo plazo, la organización, acceso y/o distribución de esos materiales.

Van de Sompel et al. (2004) expresan que debe existir un sistema de comunicación académica futura que incorpore de forma nativa las relaciones y las interacciones de la comunicación académica de la forma más detallada posible. Esto permitirá rastrear el origen de las ideas específicas y tendencias en un momento específico en el tiempo y proyectará las futuras líneas de investigación.

Tramullas & Garrido (2006) señalan que los RI se han convertido en la principal forma de publicar, preservar y difundir la información digital de las instituciones, gracias a la conjunción del acceso abierto, del software libre y de los estándares abiertos aplicados en este dominio. También señalan que eso se añade al desarrollo de las revistas y publicaciones electrónicas de calidad, que permite a las comunidades investigadoras estar actualizadas y comunicar los avances obtenidos.

Por otra parte, en la Universidad Autónoma de Madrid, Flores & Sánchez (2007) definen a un RI como un conjunto de servicios web centralizados, creados para organizar, gestionar, preservar y ofrecer acceso libre a la producción científica, académica o de cualquier otra naturaleza cultural, en soporte digital, generada por los miembros de una institución.

En el 2008 Bustos-González & Fernández-Porcel (2008) publican unas directrices para la creación de RI en universidades y organizaciones de educación superior, y definen a los RI como un archivo electrónico de la producción científica de una institución (en un formato digital) que contiene mecanismos para importar, identificar, almacenar, preservar, recuperar y exportar un conjunto de objetos digitales, normalmente, desde un portal web. Afirman que el RI es una vía de comunicación científica, pero no puede ser entendido como un canal de publicación, sino que debe comprenderse como un complemento al proceso de publicación científica formalizado con revisión por pares.

Estas definiciones han servido de base para la creación de otros repositorios similares a los institucionales, que en muchas ocasiones tienden a confundirse porque incluyen entre los objetivos principales la preservación de los recursos, el acceso irrestricto a los usuarios, la filosofía del acceso abierto, etc.

A continuación se mencionan algunos tipos de repositorios digitales con el propósito de mostrar la diversidad presente:

- *Repositorios de Datos:* referente a los datos finales de investigación (material factual registrado y/o datasets) aceptado por la comunidad científica y necesario para validar los resultados de la investigación, según el National Institutes of Health (Torres-Salinas, Robinson-García, & Cabezas-Clavijo, 2012).
- *Repositorios de Documentos Administrativos:* abarcan los documentos administrativos de la institución (Texier, De Giusti, Lira, Oviedo, & Villarreal, 2013).
- *Repositorios de Objetos de Aprendizaje:* administran recursos para la instrucción, aprendizaje o enseñanza apoyada por la tecnología (López-Gúzman, 2005).
- *Repositorios Multimedia:* abarcan materiales de audio, videos, etc (Texier, 2013).
- *Repositorios Temáticos:* tratan de la producción de un tema en particular, sin importar si pertenece a una persona o institución. Si tratan varios temas son conocidos como multidisciplinarios (CSIC, 2014; OpenDOAR, 2014; ROAR, 2014).
- *Repositorios de Tesis y Disertaciones:* solo administran ese tipo de material y algunas veces funcionan como agregador de contenido (Flores & Sánchez, 2007).

### 2.2.1. Diferencias con las bibliotecas digitales

Los Repositorios Institucionales están diseñados, principalmente, para recoger, preservar y poner a disposición la producción de una institución; a su vez, las Bibliotecas Digitales pueden estar organizadas bajo otros principios: temas, disciplinas,

o incluso tipos de documentos en particular. En siguientes líneas se mencionan algunas de las principales diferencias entre BD y RI (Texier, 2013):

- Los RI están organizados en relación con una comunidad institucional en particular, en cambio, las BD están construidas con un número diferente de posibles principios organizativos: tópico, sujeto, disciplina, etc.
- A menudo los RI dependen de la contribución voluntaria conocida como autoarchivo. Las BD son producto de una estrategia deliberada de desarrollo de colecciones por parte de los profesionales de la biblioteca, donde muchas veces tienen contenidos más allá de la propiedad institucional.
- Los RI se centran en ser solamente repositorios y ofrecer servicios limitados a los usuarios, por su parte, las BD pueden incluir aspectos de servicios como: referencia, asistencia, interpretación de contenidos, es decir, en ellas se involucra al apoyo de personal de bibliotecas en la búsqueda y diseño de información adicional.
- Los RI tienen que ser interoperables a través de estándares para la recogida de la información que administran, en cambio, las BD tienen sus propios sistemas y en muchas ocasiones no se preocupan por ser interoperables con otros sistemas.
- Los elementos de un repositorio suelen ser de origen digital, las bibliotecas por lo general tienen que digitalizar los viejos materiales (analógico).

En resumen, un Repositorio Institucional es una Biblioteca Digital y una Biblioteca Digital es un Repositorio Institucional, gracias a que actualmente ambos ofrecen servicios similares y el uso de cada uno de términos (RI y BD) depende del ámbito donde se aplica y, por ende, de los recursos con los que se desean trabajar (Xia & Opperman, 2010).

### 2.2.2. Funciones básicas de los RI

Algunas características de los RI son (Jeevan, 2004; Texier, 2013):

- Los RI deben recoger, almacenar, y, organizar la información y el conocimiento en forma digital.

- Los RI pertenecen a una institución académica o de investigación.
- Los materiales de las instituciones que representan su producción intelectual deben estar presentes en el RI, dando lugar a una colección de documentos y objetos, de varios tipos y formatos.
- Investigadores afiliados u otras personas pertenecientes a la organización pueden depositar directamente sus textos, conjuntos de datos, archivos de sonido, imágenes o cualquier otro tipo de documento, de manera personal (autoarchivo) o a través de los catalogadores.
- Los documentos pueden estar en cualquier etapa del proceso de la investigación académica, esto depende de la política de la institución sobre los documentos que se pretenden depositar.
- Fortalecen la comunicación y colaboración entre la investigación, las empresas, el gobierno y las comunidades educativas.
- Contribuyen con las oportunidades de aprendizaje permanente.

Estas características se relacionan con las funciones básicas que intervienen en la creación de un repositorio, tales como:

- Identificar y recopilar contenido a ser digitalizado.
- Clasificar y digitalizar los contenidos identificados y recopilados.
- Editar el formato y el almacenamiento de dichos contenidos.
- Diseñar motores de búsqueda de forma selectiva, accesible y que permitan la difusión de los contenidos.
- Publicar contenido digital para el acceso del usuario final.
- Garantizar la preservación del contenido recopilado e identificado.

Estas funciones implican ciertas subtareas como:

- Transformación de la información impresa a formato digital.
- Almacenamiento, recuperación y manipulación de la información electrónica.
- Conversión de datos de diversos medios o formatos a formato digital.
- Técnicas de clasificación para mejorar la eficiencia de recuperación.

- Gestión de dispositivos para el almacenamiento masivo.
- Edición y formato de los contenidos en formatos como HTML, PDF, JPEG, MPEG, etc.
- Facilidades de navegación y consultas para la recuperación de contenidos.
- Opciones para distribuir y acceder rápidamente a la información a través de comunicaciones estándares y de una infraestructura informática segura.
- Seguridad de los contenidos ofrecidos por los diferentes métodos de cifrado y cortafuegos (firewall) apropiados.

### 2.2.3. Módulos funcionales de los RI

Según las recomendaciones de la norma ISO 14721 para gestión de un sistema abierto de archivo de información (CCSDS, 2012), todo repositorio debe tener los siguientes procesos funcionales de acuerdo con el material depositado: carga, almacenamiento del archivo, gestión de datos y acceso. A partir del funcionamiento del depósito los procesos funcionales son: planeamiento de la preservación y administración (De Giusti, Lira, Villarreal, & Texier, 2012).

#### *2.2.3.1. Carga* (*Ingest*)

Está relacionado con habilitar el proceso para la recepción de los objetos digitales procedentes de diferentes vías. Dentro de este proceso se realizan actividades de catalogación para finalizar los proceso de carga. Según la norma ISO 14721 las principales funciones son:

- Proveer servicios y funciones para aceptar el paquete de información presentado (SIP) por parte de los *Productores* (o a partir de elementos internos bajo control de la administración)
- Preparar los contenidos para el almacenamiento y gestión dentro del archivo.
- Asegurar la calidad/validación de los SIP.
- Generar el AIP (paquete de información del archivo) que cumple con los estándares de formato de datos y documentos.
- Extraer la información descriptiva y enviarla al *Data Management*.
- Coordinar las actualizaciones en las entidades *Archival Storage* y *Data*

*Management*.

*2.2.3.2. Almacenamiento del archivo* (A*rchival storage*)

Se refiere a los mecanismos de preservación de los objetos digitales en un repositorio. Según la norma ISO 14721 sus funciones son:

- Proveer servicios y funciones para el almacenamiento, mantenimiento y recuperación de los AIP, es decir, recibe el AIP y lo hace persistente.
- Gestionar las jerarquías de almacenamiento.
- Configurar niveles especiales de servicio, seguridad y protección (por ejemplo, backups).
- Proveer estadísticas de inventario, capacidad disponible, etc.
- Verificar errores.
- Proporcionar un mecanismo estándar para el seguimiento y verificación de la validez de los datos.
- Proveer un mecanismo de duplicación de los contenidos en una lugar físico separado.
- Entregar el AIP a la entidad *Access* para cumplir con las solicitudes.

*2.2.3.3. Gestión de datos (Data management)*

Se encarga de la gestión y mantenimiento de los metadatos descriptivos de los objetos digitales almacenados. Está vinculado con procesos de catalogación e indexación de los recursos. Sus funciones según la norma ISO 14721 son:

- Proveer servicios y funciones para poblar, mantener y acceder a la información descriptiva.
- Administrar la base de datos (mantenimiento de las definiciones de vistas y esquemas, y la integridad referencial).
- Gestionar las actualizaciones de la base de datos, por ejemplo cargar nueva información descriptiva o datos administrativos del OAIS (norma ISO 14721).
- Ejecutar consultas de la entidad *Access* y generar los resultados con sus respectivos reportes.

*2.2.3.4. Acceso (Access)*

Proporciona servicios a los usuarios para la realización de búsquedas en el repositorio y la visualización de los materiales. Dentro de este módulo funcional se usa la indización realizada en otro módulo y ofrece los procesos de búsqueda (*search engine*) y navegación (*browsing*).

- Proporcionar servicios y funciones que dan soporte al actor Consumidor para determinar la existencia, descripción, localización y disponibilidad de la información almacenada en el OAIS.
- Permitir al Consumidor, realizar solicitudes y recibir información de los productos, es decir, coordinar la comunicación (acceso) entre el Consumidor y el OAIS. Las 3 categorías de solicitudes del Consumidor: query requests, report requests & orders.
- Aplicar controles para limitar el acceso y protección de la información.
- Generar el DIP (paquete de información de diseminación), a partir de recuperar el AIP de la entidad *Archival Storage* y la información descriptiva del *Data Management*.

*2.2.3.5. Planeamiento de la preservación (Preservation planning*)

Realiza el monitoreo y hace un seguimiento de la evolución tecnológica para definir políticas de conservación de la información almacenada. Sus funciones según la recomendaciones de la norma ISO 14721 son:

- Proporcionar servicios y funciones para monitorear el entorno del OAIS.
- Proporcionar reportes de análisis de riesgo, alertas de requisitos y estándares independientes.
- Identificar tecnologías que pueden causar obsolescencia.
- Desarrollar y recomendar estrategias y estándares para habilitar el OAIS para cambios futuros.
- Diseñar planes de migración y prototipos, para implementar políticas y directivas de administración de IPs.
- Ofrecer asistencia en el diseño y revisión para especificación de los SIP y AIP

para incorporaciones específicas.

*2.2.3.6. Administración del depósito (Administration*)

Facilita la gestión operativa del sistema. Sus funciones son:

- Proveer servicios y funciones para la operación global del sistema de archivos.
- Solicitar la información necesaria sobre los archivos y negociar los acuerdos con los Productores.
- Revisar la funcionalidad del sistema de archivos, controlar los cambios de la configuración y mantener su integridad y trazabilidad.
- Auditar las operaciones del sistema, performance y uso.
- Gestionar la configuración del sistema de hardware y software
- Desarrollar políticas de gestión de archivo por jerarquías, incluyendo políticas de migración.
- Administrar los mecanismos para restringir y permitir acceso a los elementos del archivo.
- Verificar el PDI (paquete de información de descripción de preservación) según los usos de la comunidad designada.
- Revisar periódicamente los contenidos del archivo para determinar si los datos están disponibles.
- Crear, mantener y borrar las cuentas de acceso de los consumidores.

## *2.3. Representación de Recursos (RR)*

Gracias a la aparición de los Repositorios Institucionales se ha generado un aumento en el uso de plataformas de software para RI y la necesidad de mantener dichas plataformas al mismo ritmo que otras tecnologías no específicas que rodean al repositorio. Por ello, la Representación de Recursos, dentro de este contexto, es unos de los principales problemas de este dominio. Varios autores (Candela et al., 2007; Gonçalves et al., 2004; Malizia et al., 2010; Paganelli & Pettenati, 2006) confirman tal situación y ponen en contexto del mundo científico el problema de la representación de recursos dentro de los RI, problema que se ha complejizado poco a poco,

principalmente, por la diversidad de plataformas de software y esquemas de metadatos existentes para representar los recursos de diferentes tipologías en los repositorios.

Algunas de las plataformas de software más usadas son DSpace, EPrints y Digital Commons (OpenDOAR, 2014; ROAR, 2014). No obstante, se encuentran plataformas consolidadas de desarrollos propios que abarcan un gran porcentaje de los recursos esparcidos en el mundo como: arXiv, CiteSeerX, Social Science Research Network (CSIC, 2014). Se observa que en estas diferentes plataformas se encuentran los recursos representados en diversos esquemas de metadatos. De hecho, muchos esquemas de metadatos se desarrollan de acuerdo con una variedad de usuarios y de disciplinas para facilitar la identificación, recuperación, utilización y/o gestión de recursos en los repositorios (Chan & Zeng, 2006). Por tanto, los esquemas de metadatos a través del conjunto de elementos diseñados en una estructura formal y relacionados con la semántica, la sintaxis y la obligatoriedad en sus valores, pasan a ser un elemento importante en la representación de recursos dentro de un RI. Adicionalmente, se observa en los últimos años, que la tendencia ha sido que los esquemas estén codificados en XML (Extensible Markup Language), dado que este es un estándar del W3C (World Wide Web Consortium) predominante para codificación e intercambio de datos, que están agrupados en elementos delimitados por etiquetas (Méndez, 2003a).

Los esquemas de metadatos definidos como un conjunto de propiedades que contienen valores de acuerdo con un conjunto de reglas, han sido diseñados e implementados para permitir reusos de los registros e integración de los mismos. Estos metadatos proporcionan un nivel intermedio a través del cual puedan acceder a la información, basado en los siguientes principios que los rigen (Duval, Hodgins, Sutton, & Weibel 2002): modularidad, extensibilidad, refinamiento y multilingüismo. La modularidad se considera como el principio clave de organización para caracterizar los entornos de las diferentes fuentes de contenidos de los recursos, de estilos de gestión de contenidos y de enfoques de descripción de los recursos. Este entorno modular permite combinar a los diferentes tipos de elementos de metadatos de diferentes esquemas, vocabularios y bloques de construcción para hacerlos interoperables. Es importante destacar que el contexto de un elemento en particular está limitado gracias a los *namespaces* en los esquemas de metadatos. La extensibilidad indica la posibilidad de

ajustarse a las necesidades de una aplicación en particular. El refinamiento se observa por la aplicación en diferentes dominios de acuerdo con el nivel de detalle definido por los cualificadores (granularidad) o por el conjunto de valores posibles de los elementos obtenidos por vocabularios controlados u algoritmos particulares. Finalmente, el multilingüismo permite la diversidad lingüística y cultural de estos esquemas. Estos principios de metadatos se relacionan directamente con las siguientes características de un RI (De Giusti et al., 2011), independientemente de la plataforma de software elegida:

- Complejidad del software: El software debe ser perdurable, es decir, debe estar en continua corrección de errores y generación de actualizaciones. Por tanto, entre más simple sea la representación más simple serán los modelos de datos, los procesos de carga e incluso la interfaz de usuario.
- Escalabilidad y performance: Estas dos características son proporcionales con el número de recursos. Por ejemplo, en representaciones complejas basadas en bases de datos, la complejidad de las consultas aumenta al igual que los tiempos de respuesta.
- Interoperabilidad: Definida como la capacidad que tienen algunos sistemas para intercambiar y utilizar información procedente de otro sistema diferente. La representación de información demasiado simple puede llevar a un proceso de transformación deficiente, mientras que representaciones muy complejas pueden llevar a un proceso de transformación o intercambios complicados.

### 2.3.1. Los Recursos

Los recursos son objetos físicos o digitales que se describen a partir de la enumeración de un conjunto de datos específicos (metadatos) que lo distinguen entre otros objetos (Fox, Gonçalves & Shen, 2012; Heery, 1996). Se pueden clasificar de la siguiente manera:

- Producción académica. Realizada en instituciones de educación e investigación, entre las que se encuentran: artículos de investigación, recursos educativos, tesinas de grado y tesis de postgrado, disertaciones, libros electrónicos, presentaciones.

- Producción multimedia. Relacionado con materiales como imágenes, música, audios, videos.
- Producción institucional y administrativa. Generada en instituciones privadas y públicas tales como: documentos generales, constancias, memorandos, ordenanzas, resoluciones, decretos, actas, minutas, notas, leyes, entre otros.
- Entidades abstractas. Conjunto de elementos que poseen información descriptiva propia, utilizadas en los procesos de catalogación de recursos como elementos de un vocabulario controlado. Por ejemplo: autores, instituciones, revistas y sus números, eventos y sus instancias.
- En algunas plataformas de repositorios, como DSpace, existen comunidades y colecciones, las primeras simbolizan a entidades administrativas de instituciones tales como departamentos, laboratorios, oficinas, centros de investigación, entre otros. En cambio, las colecciones simbolizan el lugar a donde pertenecen los recursos y no pueden contener otras categorías, estas deben encontrarse dentro de una comunidad.

### 2.3.2. Metadatos

El concepto de metadatos no es algo nuevo, antes de la aparición de Internet se usaba en la catalogación de libros y revistas para normalizar la información de manera que la misma se pudiera localizar y recuperar fácilmente. En el ámbito de las Ciencias de la Información, los metadatos se emplean para referirse a registros de recursos de información disponibles (Heery, 1996). Los metadatos son datos que caracterizan a otros datos, es decir, es información estructurada que describe, explica y/o localiza un recurso de información para poder identificarlo, recuperarlo, utilizarlo, administrarlo o preservarlo de una manera más clara y sistemática. Para la representación de metadatos se han desarrollado distintos modelos, esquemas, formatos o estándares, que si bien en muchos casos comparten una sintaxis y estructura de la información en XML, difieren atendiendo a los propósitos de la información que describen (Méndez, 2003b).

A continuación se exponen los esquemas de metadatos más utilizados en el área de las Ciencias de la Información (Dodero, Palomo-Duarte, & Karampiperis, 2012;

Greenberg, 2005):

- Dublin Core (Dublin Core Metadata Initiative) (DCMI, 2014). También conocido como DC es eficaz conjunto de elementos para describir recursos, cada elemento es opcional y puede repetirse. El esquema de metadatos Dublin Core conlleva dos niveles: Simple y Cualificado. El Dublin Core Simple presenta quince elementos (Contributor, Coverage, Creator, Date, Description, Format, Identifier, Language, Publisher, Relation, Rights, Subject, Source, Title y Type) y el Dublin Core Cualificado presenta elementos adicionales al Dublin Core Simple. La semántica del Dublin Core ha sido establecida por un grupo internacional e interdisciplinario de profesionales de la biblioteconomía, la informática, y de otros campos teórico-prácticos relacionados.
- MARC 21 (MAchine-Readable Cataloging - Century 21st) (MARC, 2014). Permite estructurar e identificar los datos bibliográficos (tal como títulos, nombres, temas, notas, información sobre publicación, y descripción físicas de ítems) de tal forma que puedan ser reconocidos y manipulados por computadora. Este formato fue creado en 1999 como un resultado de la combinación de los formatos MARC de Estados Unidos y Canadá. Tiene cinco clases: *Bibliographic Format, Authority Format, Holdings Format, Community Format, and Classification Data Format*.
- Metadata Object Description Schema (MODS) (MODS, 2014). Esquema para la representación de registros derivados del formato bibliográfico del MARC 21, pero usando etiquetas basadas en denominaciones textuales.
- Metadata Authority Description Schema (MADS) (MADS, 2014). Relacionado con MODS, representa el formato de autoridad según MARC 21. De este modo, permite incluir información sobre agentes (personas y organizaciones), eventos y términos (conceptos, géneros, etc.).
- Metadata Encoding and Transmission Standard (METS). (METS, 2014) Está desarrollado por Network Development y MARC Standards Office de la Library of Congress. Es un formato que registra la estructura jerárquica y contenedora de un objeto digital: nombre, archivos, ubicación, estructura y metadatos asociados. Un documento METS posee un formato estandarizado para transmisión de

metadatos que se estructuran en XML. Un documento METS consta de siete secciones: cabecera METS, metadatos descriptivos, metadatos administrativos, archivo, mapa estructural, enlaces estructurales y comportamientos.

- Preservation Metadata Implementation Strategies (PREMIS) (PREMIS, 2014). Se enfoca en estrategias de implementación de metadatos de preservación de recursos digitales. PREMIS está conformado por un grupo de trabajo internacional patrocinado por Online Computer Library Center (OCLC) y Research Libraries Group (RLG) que en el 2008, elaboró el diccionario de datos PREMIS para metadatos de preservación (PREMIS, 2008), el cual define los metadatos de preservación como información que utiliza un repositorio para dar soporte al proceso de preservación digital. El diccionario define un conjunto de unidades semánticas, de propiedades y de información que la mayoría de los repositorios necesita conocer de sus entidades para asegurar la preservación. El modelo de datos PREMIS define cinco entidades: entidades intelectuales, objetos, agentes, acontecimientos y derechos.
- Existen otros estándares de metadatos que se han desarrollado por distintos usuarios e instituciones y, que han ayudado a consolidar repositorios de datos en diferentes áreas. Algunos de los más conocidos son: Darwin Core, DDI (Data Documentation Initiative for Social and Behavioral Sciences Data), DIF (Directory Interchange Format for Scientific Data), EML (Ecological Metadata Language), NBII (National Biological Information Infrastructure), TEI (Text Encoding Initiative), ETD (Electronic Theses and Dissertations).

Otros estudios presentan mapas visuales sobre la diversidad de estándares de metadatos para diferentes áreas, por ejemplo, Riley (2010) realiza una investigación sobre el patrimonio cultural en el que se evidencian 105 estándares de metadatos y expone las relaciones sobre la base de cuatro categorías: comunidad, dominio, función y propósito.

La diversidad de esquemas de metadatos existentes junto con los recursos se convierten en el pilar central del diseño de software para repositorios. Se ha observado como los recursos son muy variados en cuanto a su tipología, lo que modifica

considerablemente su representación y tratamiento (esquema de metadatos, vocabularios controlados, etc.). Por ello, el almacenamiento físico debe realizarse cuidadosamente para asegurar: la recuperación en forma eficiente, la preservación en el tiempo y las capacidades de interoperabilidad con otros repositorios de los recursos a través de sus metadatos. Por tanto, la representación de recursos dentro de un repositorio institucional es compleja para el manejo de los recursos de una forma clara y transparente.

A continuación, se presentan algunas características básicas que se deben tomar en cuenta para una representación de los recursos en repositorios:

- Los recursos deben estar catalogados en más de un esquema de metadatos a fin de evitar pérdida de información cuando se almacenen y se realicen mapeos entre los esquemas de metadatos deseados.
- Debido a la gran diversidad en las estructuras (planas y jerárquicas) y restricciones (simples y complejas) de cada esquema de metadatos (formatos), es necesario plantear una solución de representación clara, flexible, factible, escalable, interoperable, sostenible y que no represente un desafío de configuración para los administradores, así como tampoco sea complicada su utilización por los usuarios.
- Muchos campos presentes en los esquemas de metadatos necesitan ser estandarizados (por ejemplo: aplicación de vocabularios controlados o tesauros) para garantizar una identificación y recuperación correcta de acuerdo con las características de los recursos usados, a fin de evitar la redundancia y garantizar la integridad de la información.
- En estos sistemas, frecuentemente, se encuentran elementos que poseen información descriptiva propia, conocidos como entidades abstractas que son representadas de forma separada. Por ello, es necesario permitir su reutilización en los diferentes módulos del repositorio y garantizar su representación.
- Las entidades abstractas pueden tener distintas representaciones según el esquema de metadatos en el que se represente el recurso y la semántica deseada en dicha representación.
- Todos los metadatos de los recursos de un repositorio institucional deben

almacenarse de forma persistente, ante lo cual el almacenamiento debe ser configurable a cualquier tipo de paradigma de base de datos e independiente del modelo de representación de los recursos definido.

- Una infraestructura de software que garantice el intercambio de información con otros repositorios y, a su vez, asegure tiempos de respuesta bajos con capacidades escalables.

### 2.3.3. Factores que inciden en la representación de recursos

El trabajo de Texier & De Giusti (2014) descripto en una revisión de literatura describió un corpus que expone los problemas relacionados con la representación de recursos (RR), a saber: diversidad de soluciones tecnológicas, tratamiento de diferentes tipologías de recursos, esquemas de metadatos, recomendaciones de almacenamientos de recursos, la preservación de recursos, la recomendaciones del modelo OAIS y la aplicación de modelos conceptuales para solucionar el problema de la RR como un todo tales como el modelo FRBR (*Functional Requirements for Bibliographic Records*), FRAD (*Functional Requirements for Authority Data*) y FRSAD (*Functional Requirements for Subject Authority Data*) (Altenhöner, 2006; Brindley, Muir, & Probets, 2004; Jeevan, 2004; Mischo, Norman, Shelburne, & Schlembach, 2007; Weng & Mi, 2006; Zavalina, 2012).

A partir del corpus se pudieron determinar los factores que inciden en una representación de recursos, los cuales se agruparon en los siguientes ejes de análisis:

1. *Recursos:* el trabajo de Altenhöner (2006) menciona que los objetos digitales deben verse y tratarse como bitstreams, según la norma ISO, por el contrario, en los trabajos de Weng & Mi (2006) y Mischo et al. (2007) estudian la diversidad de recursos culturales y electrónicos, adaptados a esquemas de metadatos conocidos como MARC o METS. De igual manera, Jeevan (2004) explica el tratamiento de recursos académicos y científicos digitalizados y Brindley et al. (2004) se centran en la migración del recurso libro. En cambio, Zavalina (2012) estudia los recursos en forma general y adaptado a la familia de modelos FRBR. En resumen, se identifican un gran número de recursos que deben ser parte de un

RI, tales como: artículos, revisiones, proceedings, monografías, tesis, datasets, documentos administrativos, documentos de gobierno, reportes técnicos, etc. De esta manera un RI tiene que adaptarse a las tipologías existentes y a los nuevos tipos de recursos que se generan con el paso del tiempo.

2. *Esquemas de metadatos:* en el corpus precedente se evidencian varios esquemas tradicionales como soluciones a los RI. Entre los más mencionados está el esquema METS estudiado por Altenhöner (2006) y Brindley et al. (2004); y el MARC por Weng & Mi (2006) y Mischo et al. (2007). Adicionalmente, los autores Altenhöner (2006), Mischo et al. (2007), Jeevan (2004) y Brindley et al. (2004) usan esquemas de desarrollo propio y de propósito general, por ejemplo Dublin Core. Mención especial merece el trabajo Brindley et al. (2004) porque expone el tema de la preservación digital y recomienda el uso del esquema PREMIS. Todos estos esquemas deben ser permitidos por los RI para evitar la pérdida de información y hacerlos interoperables con otros RI. También los repositorios deben ser capaces de adaptarse a futuros esquemas de metadatos que se establezcan por recomendaciones.
3. *Almacenamiento:* el trabajo de Jeevan (2004) recomienda cómo debe ser el proceso para gestionar la persistencia de los metadatos y el objeto digital, y Brindley et al. (2004) trabajan directamente con el paradigma de base de datos relacional. Cabe destacar que los trabajos de Altenhöner (2006) y Brindley et al. (2004) mencionan las recomendaciones de la norma ISO 14721. Por ello, estos trabajos se enfocan en solucionar la persistencia de la información a través del modelo y en garantizar la recuperación de la información de forma normal (a través de consultas) y ante desastres no previstos (a través de copias espejo o distribuidas). Asimismo, recomiendan el uso de identificadores persistentes de recursos y la ejecución de tareas de revisión y modificación de los recursos para mejorar la integridad y calidad de la información, ya que los depósitos provienen de diferentes fuentes y vías y, del uso correcto de controles bibliográficos.
4. *Catalogación:* se recomiendan diferentes vías para garantizar la normalización de la información ingresada y almacenada dentro de un RI. Weng & Mi (2006) se centran en la recomendación del modelo FRBR para catalogar los diferentes

tipos de recursos, Mischo et al. (2007) sugieren el uso estricto de controles bibliográficos como el análisis documental formal y de contenido, y el trabajo de Jeevan (2004) recomienda el uso de una guía para catalogar basada en los principios básicos de la disciplina. Asimismo, Zavalina (2012) sugiere el uso de los modelos de la familia FRBR (FRBR, FRAD y FRSAD) y la aplicación del código de catalogación RDA (*Resource Description and Access*) para no usar las AACR2 (segunda edición de las reglas de catalogación *Anglo-American-Cataloguing*) o sus predecesores, y Zavalina (2012) enfoca el problema de la catalogación a través del acceso por materia a partir de dos actores: los usuarios que buscan la información en los sistemas de repositorios y los profesionales de la información que analizan y crean los metadatos de los recursos. Por tanto, estos trabajos se encargan de facilitar los puntos de acceso para que los usuarios localicen y recuperen la información indizada, que por lo general, se centra en los campos título, autor y materia, y en algunos casos, permite la búsqueda a texto completo.

5. *Incorporación de recursos*: basado en el proceso funcional de ingreso de ítems a un repositorio, el trabajo de Altenhöner (2006) señala que los depósitos provienen de diferentes fuentes y se realizan por diversas vías, pero no se enfoca en una solución con base en alguna norma, como sucede en el trabajo de Brindley et al. (2004) quienes recomiendan mantener los criterios de la norma ISO y el uso de los paquetes de información, que según la norma se denominan SIP. Situación similar ocurre en el trabajo de Jeevan (2004) que solamente explica la importancia del proceso de incorporación pero no lo menciona como un problema ni una ventaja, simplemente como uno de los pasos a seguir para hacer disponible los recursos para una comunidad.
6. *Gestión de los datos:* para poblar, mantener y acceder a la información los trabajos hallados remiten a, en el caso de Weng & Mi (2006) en la aplicación del modelo FRBR y los trabajos de Altenhöner (2006) y Mischo et al. (2007) usar guías de catalogación para el control de la información. Zavalina (2012) recomienda que el manejo de los datos se centralice sobre la familia de modelos FRBR. En síntesis, estos trabajos se enfocan en poder crear los puntos de acceso

y de normalizar la información (metadatos) de los objetos digitales ante el crecimiento de los repositorios, procesos de migración o el simple uso.

7. *Acceso:* para el usuario el acceso es importante ya que representa la puerta de entrada al repositorio. Los trabajos encontrados estudian la problemática del acceso a partir de los procesos funcionales *Indexing*, *Search* y *Browsing*. En los trabajos de Weng & Mi (2006) y de Zavalina (2012) se estudia una solución al acceso a través de la aplicación del modelo FRBR, la cual ayuda a elaborar una buena catalogación, permitiendo que la indización de los recursos sea correcta y garantice resultados correctos para las búsquedas y navegación por los diferentes puntos de acceso definidos. Sin embargo, Zavalina (2012) incorpora la importancia de la gestión de las materias para un correcto acceso e incorpora los modelos FRAD y FRSAD. Los trabajos de Mischo et al. (2007) y Jeevan (2004) dan pautas generales en el acceso web, pero no plantean soluciones concretas al problema.

## *2.4. Estándares a considerar en los RI*

### 2.4.1. XML

XML es un lenguaje de marcas desarrollado por el *World Wide Web Consortium* (W3C), conocido en inglés como *eXtensible Markup Language (W3C, 2014a)*. Deriva del lenguaje SGML (ISO 8879) y permite definir la gramática de lenguajes específicos para estructurar documentos electrónicos de gran escala. Juega un papel importante en el intercambio de datos en la red. XML se creó para que fuera igual a la forma de recibir y procesar la información del HTML, extensible y fácil de implementar, programar y aplicar en los distintos sistemas.

XML es un lenguaje que permite describir los contenidos dentro del propio documento, reutilizar sus partes, jerarquizarlos y estructurarlos. La información estructurada presenta varios contenidos (texto, imágenes, audio, etc.) y formas (hojas de cálculo, tablas de datos, parámetros de configuración, etc). La forma indica qué papel puede jugar el contenido. Por ejemplo, la codificación en *Dublin Core* en un esquema

XML permite que los motores de búsqueda puedan encontrar información de forma mucho más eficiente, permitiendo el intercambio entre diversos sistemas por presentar un formato estándar para representarla sobre un tema específico.

### 2.4.2. OAI-PMH

El protocolo OAI-PMH (*Open Archives Initiative-Protocol Metadata Harvesting*) realiza el intercambio de información para que desde diversos puntos (proveedores de servicio) se puedan hacer búsquedas que abarquen la información recopilada en distintos repositorios asociados (proveedores de datos), como se observa en la Figura 2-1 (Rusch-Feja, 2002).

El volumen y crecimiento de archivos y de motores de búsqueda han hecho que se recurra a los metadatos y al lenguaje de marcas XML para conseguir la interoperatividad entre archivos. Esta iniciativa permite el *harvesting* de los metadatos, entendido por la localización y recopilación de los metadatos que describen los registros contenidos en los repositorios mediante el protocolo OAI-PMH (OAI, 2014). Los metadatos a intercambiar deben codificarse en *Dublin Core Simple* (no calificado).

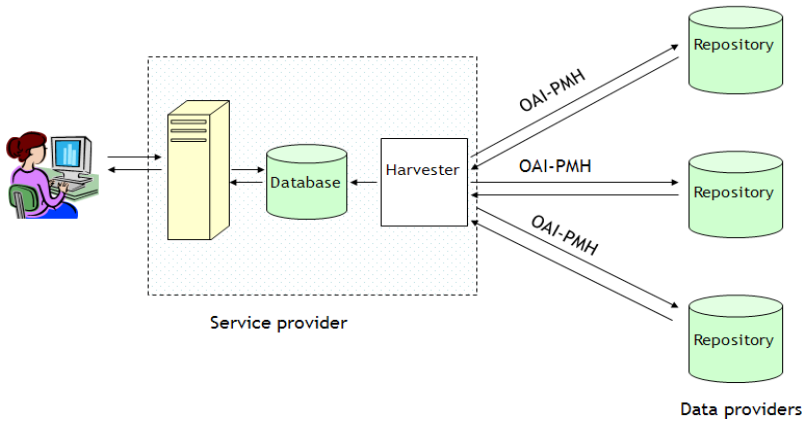


*Figura 2-1. Proveedores del Protocolo OAI-PMH*

2.4.2.1. *Proveedores de datos*: son los depósitos de documentos que proporcionan los registros de los recursos que almacenan. Un proveedor de datos ofrece los metadatos a través de un archivo XML siguiendo una estructura específica de los recursos resguardados (acceso público o privado).

2.4.2.2. *Proveedores de servicios*: son conocidos como recolectores o agregadores de

contenidos y se definen como un programa encargado de recolectar la información ofrecida por los proveedores de datos con el objetivo de incorporarles algún valor añadido y presentarlos a los usuarios finales.

### 2.4.3. SWORD

SWORD (*Simple Web-service Offering Repository Deposit*) es un estándar de interoperabilidad que permite a los repositorios digitales aceptar el depósito de los contenidos de múltiples fuentes en diferentes formatos por ejemplo, documentos XML (Jones, 2013). De la misma manera que el protocolo HTTP permite que cualquier navegador web para hablar con cualquier servidor web, SWORD permite a los clientes hablar con los servidores de repositorios (De Giusti, Villarreal, Terruzzi, Oviedo & Lira, 2013).

### 2.4.4. OpenSearch

OpenSearch (2014) es un conjunto de tecnologías que logra la publicación de los resultados de búsqueda en un formato adecuado para la sindicación y agregación. Es una manera para que los sitios web y motores de búsqueda publiquen los resultados de búsqueda en un formato estándar y accesible. *OpenSearch* 1.0 fue publicado en marzo de 2005. Consiste en cinco pilares: archivos descriptivos, sintaxis de consulta, RSS, agregadores y un "Auto-discovery".

### 2.4.5. RSS

RSS (*Really Simple Syndication*) es un formato XML para compartir contenido en formatos de fuentes web utilizadas para publicar, frecuentemente, actualización de productos de software, tales como entradas de *blogs*, titulares de noticias, audio y video en un formato estandarizado (RSS, 2014). Un documento RSS incluye un resumen o texto completo, además de los metadatos como fechas de publicación y autoría.

Existen editores y lectores de RSS. Los editores distribuyen el contenido de forma automática y los lectores, a través de suscripciones vía URI, permiten a los

usuarios evitar, manualmente, la inspección de todos los sitios web que están interesados, de tal manera que todo el contenido nuevo se encuentra en el programa lector, principalmente el navegador (Cold, 2006).

### 2.4.6. XMI

XMI (XML Metadata Interchange) es un estándar de la OMG y una especificación basada en XML (OMG, 2014). XMI es la base para conseguir interoperabilidad entre herramientas de apoyo del mundo Model-Driven. Una de las grandes ventajas de XMI es que sobre él se pueden almacenar todos los modelos basados en MOF/Ecore utilizando un mismo esquema. También tiene inconvenientes, por ejemplo, el complejo formato para ser comprendido por humanos o que muchas herramientas que trabajan con XMI no se ajustan a las especificaciones de la recomendación.

### 2.4.7. XSD

XSD (XML Schema Definition) es un lenguaje que sirve para expresar restricciones sobre documentos XML (W3C, 2014b), utilizando componentes del esquema para limitar y documentar el significado, el uso y las relaciones de sus partes: tipos de datos, elementos, contenidos, atributos y valores.

### 2.4.8. MOF

Meta-Object Facility (MOF) define un subconjunto de UML para describir conceptos de modelado de clases dentro de un repositorio de objetos, es decir, usa los mismos conceptos y la misma sintaxis concreta de los diagramas de clases UML. Es un estándar de la OMG para la ingeniería conducida por modelos. Su arquitectura se define en términos de sí mismo y cada elemento de un modelo en cualquiera de las capas debe tener una correspondencia estricta con un elemento del modelo de la capa inmediatamente superior (Steinberg, Budinsky, Merks, & Paternostro, 2008).

### 2.4.9. Ecore

Es el metamodelo usado para representar modelos en EMF, también es conocido como el meta-metamodelo, es decir, es el corazón del modelado en Eclipse. Fue la implementación del MOF en Eclipse, pero realizada en Java. Un metamodelo es simplemente el modelo del modelo y si un modelo es en sí mismo un metamodelo, entonces, el metamodelo es en efecto un meta-metamodelo (Steinberg et al., 2008).

MOF y Ecore tienen muchas similitudes en su capacidad para especificar las clases y sus características estructurales y de comportamiento, la herencia, y los paquetes. Se diferencian en el área de ciclo de vida, las estructuras de tipos de datos, las relaciones del paquete y los aspectos complejos de las asociaciones. Tanto Ecore como MOF se encuentran en la capa superior de la arquitectura de 4 capas MDA (Gerber & Raymond, 2003).

### 2.4.10. EMF

La plataforma de desarrollo de software Eclipse está dividida en numerosos proyectos de alto nivel: Eclipse Project, Modeling Project, Tools Project y Technology Project. El Eclipse Modeling Framework (EMF) es el framework de modelado y generación de código que explota todos los recursos (a través de dichos proyectos) que ofrece Eclipse para construir herramientas y otras aplicaciones basadas en modelos estructurados, donde los modelos están basados en formatos XMI (Steinberg et al., 2008).

### 2.4.11. EMP

El Eclipse Modeling Project es un proyecto relativamente nuevo de nivel superior en Eclipse. El núcleo del proyecto, EMF, ha estado en existencia durante todo el tiempo de la plataforma Eclipse. El proyecto de modelado Eclipse es en gran medida un conjunto de proyectos relacionados con las tecnologías de modelado y del desarrollo de software dirigido por modelos. Esta colección se formó para coordinar y enfocar las capacidades de desarrollo de software dirigido por modelos dentro de Eclipse (Gronback, 2009).

# CAPÍTULO 3

# EL MODEL-DRIVEN

La construcción de una propuesta de software conlleva a desarrollarla bajo una metodología que permita replicarla, verificarla, validarla y probarla, por personas externas al mismo, así como también permita generar una serie de productos a fines al software desarrollado. Por ello, para el marco de referencia propuesto en este estudio se selecciono el enfoque de Desarrollo de Software Dirigido por Modelos, también conocido como Model-Driven.

En vista de la diversidad de paradigmas para la construcción de software, la comunidad de desarrolladores de software se ha sensibilizado respecto de la necesidad de formalizar la construcción de componentes de software y guiar tales desarrollos a través de modelos definidos. Tales modelos son un conjunto de elementos que sirven para demostrar la consistencia de una teoría, es decir, representan con detalle un sistema dado (Whittle, Clark & Kühne, 2011). Este enfoque es conocido como Model-Driven, se define como una metodología de desarrollo de software que permite aplicar las ventajas del modelado en actividades de ingeniería de software (Brambilla, Cabot, & Wimmer, 2012). Comprende los siguientes aspectos (Brambilla et al., 2012):

- Conceptos. Componentes sobre los que se construyen los modelos y las transformaciones.
- Notación. La forma en que se representan los conceptos, es decir, los lenguajes de modelado. De manera similar a los programas computacionales que necesitan los lenguajes de programación.
- Procesos y reglas. Las actividades que llevan a generar un producto final, las reglas para su coordinación y control, y las aserciones sobre las propiedades deseadas de los productos o del proceso.
- Herramientas. Aplicaciones que facilitan la ejecución de actividades o de su coordinación. En este caso serán los Entornos de Desarrollo Integrados o IDE (*Integrated Development Environments*) que se usan para la definición de los modelos y las transformaciones, y diseños de compiladores o intérpretes para

ejecutarlos y producir los artefactos de software finales.

## *3.1. ¿Qué es el Model-Driven?*

El reto para comprender el área del Model-Driven (Figura 3-1) se resume en la cantidad de acrónimos que se usan como referencia (Brambilla et al., 2012), estos pueden parecer sinónimos básicos y confusos, pero cada uno de ellos representa una subárea específica:

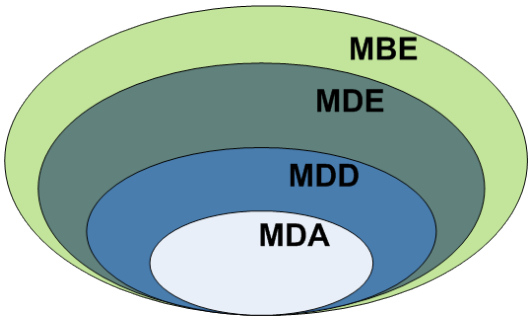


*Figura 3-1. Relaciones entre los diferentes acrónimos del Model Driven Star. Fuente: Brambilla, 2012*

- MBE (Model-Based Engineering): Tom DeMarco (1979) introdujo el concepto, relacionado con la construcción de un sistema de software precedida por la construcción de un modelo tal y como se realiza en otros del área de la ingeniería en general.
- MDE (Model-Driven Engineering): todas las variantes de "Model Driven Lo-que-sea" (*Model Driven Whatever*) son frecuentemente referidas como el acrónimo MD* (*Model Driven star*). A pesar de la enorme variedad de acrónimos que se pueden encontrar en la literatura, MDE puede ser visto como el superconjunto de todas estas variantes, por ello, cualquier enfoque MD*E podría estar bajo el paraguas MDE (Brambilla et al., 2012). Por ejemplo: MDSE (Model Driven Software Engineering), MDPE (Model Driven Product Engineering), MDWE (Model Driven Web Engineering), entre otros.
- MDD (Model-Driven Development): paradigma de construcción de software cuyas motivaciones principales son la independencia de los productores a través de estandarizaciones y portabilidad de los sistemas (Pons et al., 2010). El objetivo de MDD es separar el diseño del sistema de la arquitectura de las

tecnologías, para que puedan ser modificados independientemente. Existen dos propuestas concretas muy conocidas y utilizadas bajo este enfoque, MDA y el Modelado Específico del Dominio (DSM acrónimo inglés de *Domain Specific Modeling*) acompañado por los lenguajes específicos del dominio (DSLs acrónimo inglés de *Domain Specific Language*). Estas dos propuestas están fuertemente conectadas con los conceptos básicos MDD. Por tanto, MDA tiende a enfocarse en lenguajes de modelado basados en estándares y DSM utiliza otras notaciones no estandarizadas para definir sus modelos (Pons et al., 2010).

- MDA (Model-Driven Architecture): propuesta de MDD definida por Object Management Group (OMG). En ocasiones, el término MDA se intercambia con el de MDD, ya que MDA se refiere a las actividades que llevan a cabo los desarrolladores basado en estándares, mientras que MDD se enfoca en su definición formal (Navarro et al., 2011).

## *3.2. Ciclo de vida del MDD*

El ciclo de vida de desarrollo de software en MDD (Figura 3-2) basa su funcionalidad en tres modelos (OMG, 2003): un **modelo computacional independiente** (Computation Independent Model o CIM) que es una vista del sistema que no muestra detalles de su estructura, conocido como el modelo del dominio; corresponde, tradicionalmente, a las etapas de captura de requisitos y análisis (Pons et al., 2010). El **modelo independiente de la plataforma** (Platform Independent Model o PIM) definido a través de un lenguaje específico para el dominio en cuestión e independiente de cualquier tecnología. El modelo PIM puede traducirse en uno o más **modelos específicos de la plataforma** (Platform Specific Model o PSM). Las implementaciones de los PSM, están basados en lenguajes específicos del dominio o lenguajes de propósito general como Java, C, base de datos relacionales (SQL), Python, etc. Definidos cada uno de estos modelos, se realiza la implementación del código fuente (Implementation Model o IM) a partir de cada uno de los PSM desarrollados. Es importante destacar que los PIM o PSM pueden contener varios modelos correspondientes a distintos puntos de vista del sistema.

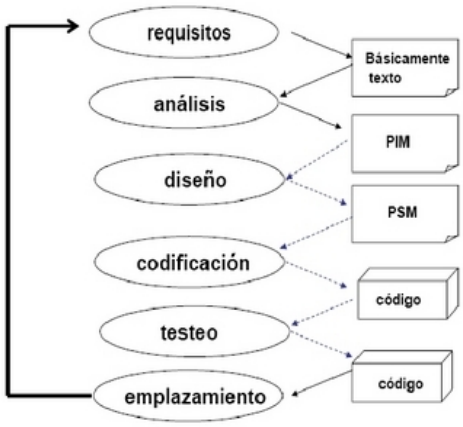


*Figura 3-2. Ciclo de vida de MDD. Fuente: (Kleppe, Warmer, & Bast, 2003)*

## *3.3. Transformaciones en MDA*

MDA plantea una metodología basada en la generación de modelos (PIM y PSM), los cuales pueden transformarse sucesivamente en otros modelos hasta obtener una representación final (usualmente código ejecutable) gracias a la definición de un lenguaje, que proporciona la posibilidad de describirlos de forma adecuada. Este lenguaje está a su vez definido a partir de un metalenguaje (Navarro et al., 2011), lo cual genera la idea de sucesivas capas recursivas (Figura 3-3).

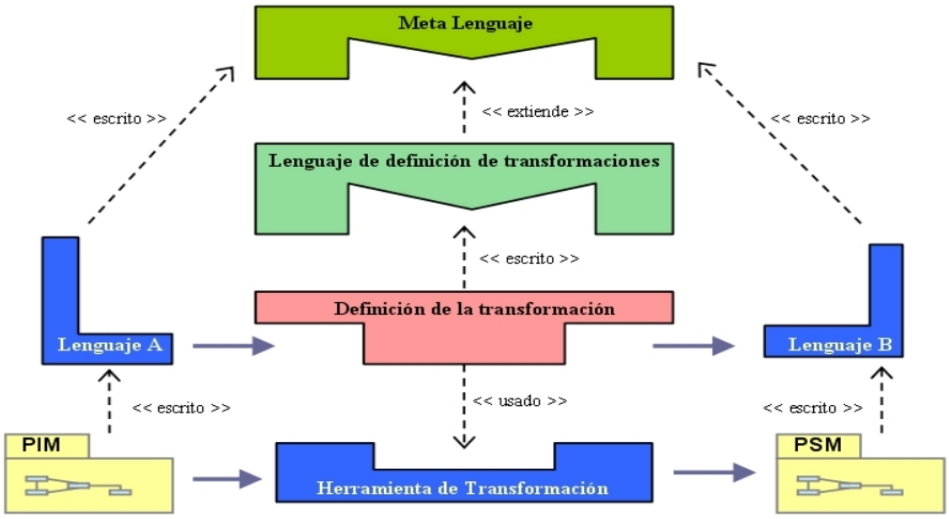


*Figura 3-3. Esquema de la Transformación. Fuente: Kleppe et al. (Kleppe et al., 2003)*

La OMG (2003) define a las transformaciones como el proceso de convertir un modelo en otro del mismo sistema mediante un conjunto de reglas. Estas reglas no

deben producir ambigüedad para garantizar una transformación única. Por ello, se puede hablar de la transformación de un modelo fuente a un modelo objetivo o destino (Figura 3-4).

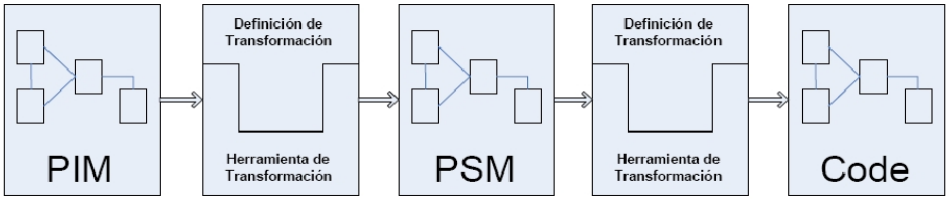


*Figura 3-4. Transformaciones en MDA. Fuente: Pons et al. 2010*

## *3.4. Arquitectura de metamodelado de 4 capas en MDA*

El OMG especifica una serie de capas o niveles (Figura 3-5) de modelado (OMG, 2003):

- Capa M3: Metametamodelo. Define los elementos existentes en la capa M2, mediante instancias de elementos existentes en esta capa. En definitiva, en este nivel está definido el meta-metalenguaje o metametamodelo. Por ejemplo, MOF y Ecore.
- Capa M2: Metamodelo. Este nivel contiene los elementos del lenguaje de modelado o metamodelo. Por ejemplo: metamodelos de UML y de Java.
- Capa M1: El modelo del sistema. A este nivel se define el modelo del sistema o la aplicación propiamente dicha. Por ejemplo, un modelo en UML o un modelo de clases Java.
- Capa M0: Las instancias. Es la capa en donde se ejecuta el sistema, donde están las instancias reales que se han creado durante la ejecución. Por ejemplo, un objeto UML o instancias de las clases Java.

Por tanto, todo este framework que propone el paradigmas Model-Driven, permite una serie de beneficios que facilita la construcción de un marco de referencia que puede ser replicado en las diferentes plataformas de software existentes, ya que la representación de recursos debe ser general y a través de los metamodelos y modelos, se pueda adaptar a los diversos repositorios institucionales.

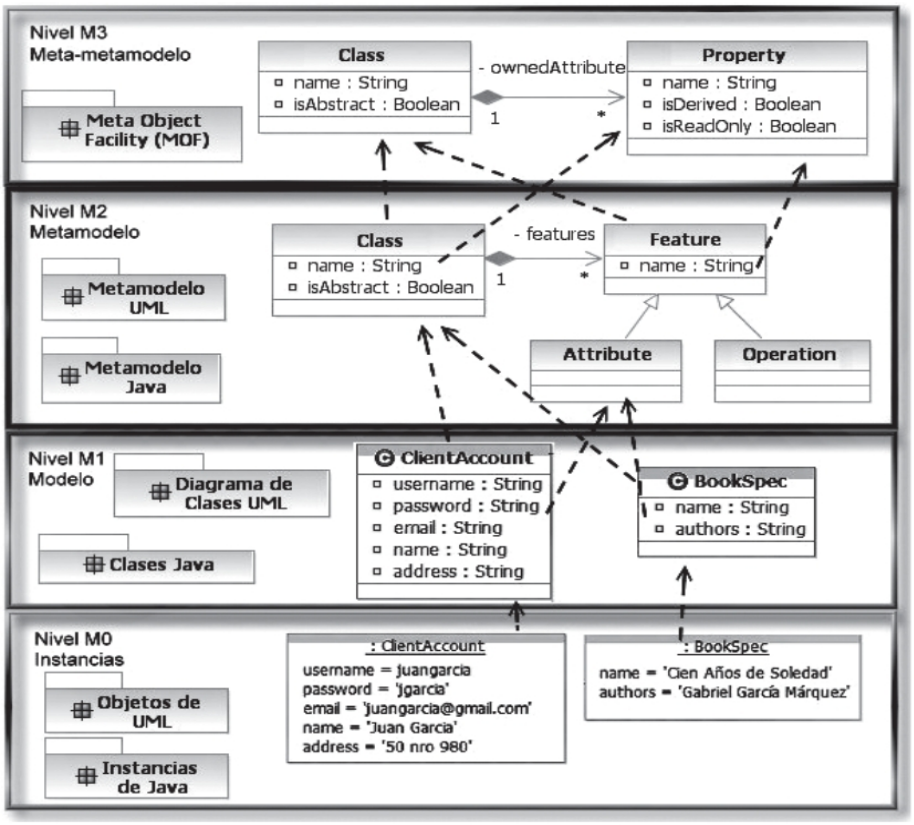


*Figura 3-5. Vista general de las relaciones entre los 4 niveles. Fuente: Pons et al. 2010*

## *3.5. Modelado específico del dominio (DSM)*

El DSM (Domain-Specific Modeling) es un tipo de metodología de la ingeniería de software conocida por crear modelos para un dominio específico a través de un lenguaje focalizado. Permiten el diseño y la implementación de sistemas computacionales con un alto nivel de abstracción (Fowler, 2010). Un DSL (Domain Specific Language) es un lenguaje de programación especializado para un dominio particular de aplicación usando conceptos del dominio seleccionado (Fowler, 2010). Estos lenguajes son frecuentemente gráficos pero también pueden ser textuales. Los productos finales son generados desde estas especificaciones de alto nivel. La automatización es posible si ambos, el lenguaje y los generadores, se ajustan a los requisitos de un único dominio (Pons et al., 2010).

## *3.6. Enfoque MDE*

MDE se puede aplicar en diferentes niveles de abstracción para poder proporcionar una

visión integral en el desarrollo del sistema de software. MDE busca soluciones de acuerdo con dos dimensiones (ver Figura 3-6): conceptualización (eje de las columnas) e implementación (eje de las filas).

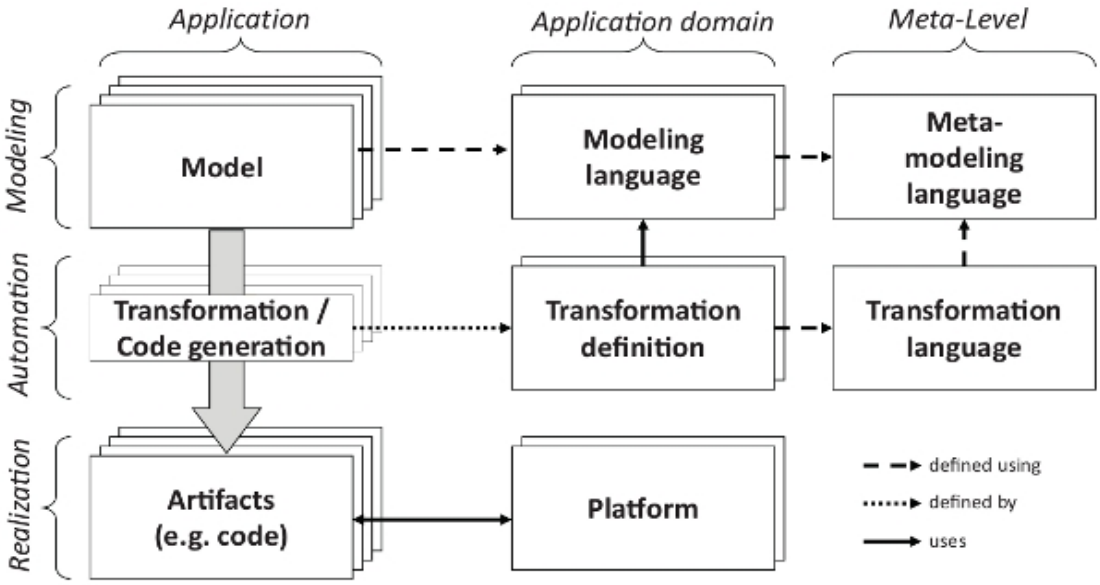


*Figura 3-6. Visión general de MDE. Fuente: Brambilla, 2012*

La implementación se observa con el mapeo de los modelos de funcionamiento de algunos sistemas existentes o futuros. De acuerdo con la Figura 3-6, se definen tres aspectos fundamentales (Brambilla et al., 2012):

- El nivel de modelado: donde se definen los modelos.
- El nivel de automatización: donde se ponen las asignaciones de la modelización a los niveles de realización en su lugar.
- El nivel de realización: donde las soluciones se implementan a través de artefactos que se encuentran actualmente en uso dentro de los sistemas de funcionamiento.

La conceptualización está orientada a la definición de los modelos conceptuales para describir la realidad. De acuerdo con la Figura 3-6, pueden aplicarse en tres niveles principales (Brambilla et al., 2012):

- El nivel de aplicación: donde se definen los modelos de las aplicaciones, las reglas de transformación que son llevadas a cabo, y los componentes de funcionamiento que se generan.

- El nivel de dominio de aplicación: donde se definen el lenguaje de modelado, las transformaciones y las plataformas de aplicación para un dominio específico.
- El meta-nivel: donde la conceptualización de los modelos y de las transformaciones son definidas.

## *3.7. Ingeniería web y MDWE*

La Ingeniería Web es una disciplina que propone extensiones y adecuaciones a los métodos y modelos tradicionales, incluye un proceso que permite crear, implementar y mantener aplicaciones web de calidad. El desarrollo de este tipo de aplicaciones se ha incrementado con el auge de Internet, volviéndose un desafío para los ingenieros de software (Pressman, 2002). A raíz de esto se crearon enfoques y metodologías tomando en cuenta el aporte tecnológico. Por ello, nace el modelado conceptual de aplicaciones web, más específicamente el Model Driven Web Engineering (MDWE), que se encuentra dentro del contexto MDE (Brambilla et al., 2012). En MDWE se pueden identificar 5 grupos (Liddle, 2011):

1. *Enfoques orientados a los datos:* tales como RMM, WebML y Hera. Tienen su origen en sistemas de base de datos y se centran en aplicaciones web de datos intensivos.
2. *Métodos orientadas al hipertexto:* tales como HDM, HDM-lite, WSDM y W2000, originados de trabajos en diseños de hipermedia y manejo de la naturaleza hipertextual de las aplicaciones web.
3. *Enfoques orientados a objeto:* siguen la tradición del modelo orientado a objetos y se incluyen los métodos OOHDM, UWE, OOWS y OO-H.
4. *Métodos orientados al software:* toman el enfoque tradicional de desarrollo de software. Web Application Extension (WAE) y su extensión WAE2 ejemplifican este enfoque.
5. *Métodos orientados al MDE:* explícitamente tomado del enfoque model-driven para el desarrollo de aplicaciones web y hacen hincapié en la generación de código fuente a partir de modelos de aplicaciones web. Ejemplos de esta categoría incluye Webile, WebSA, MIDAS y Netsilon.

### 3.7.1. WebML

Es una metodología de modelado web orientada a datos (Acerbis et al., 2008; Ceri, Fraternali, & Bongio, 2000), que se encuentra dentro de la definición MDWE y corresponde a MDE. En esta metodología se definen 4 modelos que acompañan al proceso del desarrollo del software: datos, hipertexto, gestión de contenidos e hipertexto avanzado (Ceri et al., 2003; Ceri, Fraternali, & Matera, 2002). En la propuesta de investigación presentada enfocó sólo en los modelos de datos y de hipertexto en WebRatio por la practicidad a la hora de implementar una aplicación.

- *Modelo de datos:* es una adaptación de los modelos conceptuales para el diseño de datos, similares a los que se utilizan en otras disciplinas como el diseño de bases de datos, ingeniería de software y la representación del conocimiento. Por ello, es compatible con el modelo entidad-relación usado en el diseño de bases conceptuales y de los diagramas de clases UML. Los elementos fundamentales de los modelos de datos son entidades (contenedores de elementos de datos) y relaciones (conexiones semánticas entre ellas). Las entidades tienen propiedades, llamadas atributos, con un tipo asociado. Las entidades se pueden organizar en jerarquías de generalización y las relaciones pueden ser restringidas por medio de la cardinalidad. Las instancias de entidades se consideran individualmente direccionables por medio de un identificador único (OID).
- *Modelo de hipertexto*: El objetivo es especificar la organización de las interfaces del front-end de una aplicación web. Para ser eficaz, esta especificación debe ser capaz de transmitir de una manera sencilla e intuitiva aspectos como la división lógica de la aplicación en módulos de nivel superior, donde cada uno contiene un conjunto de funciones dirigidas a un grupo específico de usuarios. Los elementos fundamentales para este modelo son: páginas, unidades y enlace, todos ellos modularizados en construcciones *areas* y *site views*. El modelado de hipertexto especifica la composición y la navegación del sitio. La composición describe qué páginas componen el hipertexto y, qué unidades de contenido constituyen una página. Las páginas del sitio web son las contenedoras de información, efectivamente, entregadas al lector y, las unidades son los elementos de contenido atómicos utilizados para publicar la información descrita

en el modelo de datos. Esta metodología define siete tipos de unidades que pueden componer páginas: data, multi-data, index (y sus variantes multichoice y hierarchical), entry y scroller. La navegación del sitio se especifica a través de enlaces que pueden ser definidos entre las unidades dentro de una única página, entre las unidades situadas en diferentes páginas, y entre las páginas. La información transportada a lo largo de un enlace se llama contexto de navegación. Los enlaces que llevan la información de contexto se llaman enlaces contextuales, mientras que los enlaces que no tienen información de contexto asociada se llaman enlaces no contextuales.

### 3.7.2. WebRatio

El desarrollo de aplicaciones con base en WebML es asistido por la herramienta WebRatio, una herramienta comercial (con una licencia para desarrollo personal y el desarrollo académico) para el diseño y la implementación de aplicaciones web, y está soportado por el IDE Eclipse (WebRatio, 2013). Los requisitos se expresan a través de un modelo de alto nivel y el código de la aplicación se genera automáticamente (mediante el uso de reglas). El código resultante es una aplicación Java basada en estándares web, y consta de tres pasos básicos: construcción de los modelos (en particular, el modelo de la aplicación se realiza utilizando WebML), personalización de las reglas y generación de la aplicación (Brambilla & Fraternali, 2013). WebRatio también soporta los estándares BPM y el más reciente estándar IFML aprobado por la OMG en el 2013.

La arquitectura de WebRatio (Figura 3-7) consiste en dos capas: una capa de diseño que provee funciones para la edición visual de las especificaciones y una capa de ejecución (run-time) que implementa los servicios básicos para la ejecución de las unidades WebML sobre un framework de aplicaciones web estándar (Brambilla, Comai, Fraternali, & Matera, 2008).

La capa de diseño, el generador de código y la capa de ejecución tienen una arquitectura de plugin: los nuevos componentes de software pueden ser acoplados con descriptores XML y hacerlos disponibles para la capa de diseño como las unidades WebML personalizadas. El generador de código se puede ampliar con las reglas XSL

para producir el código necesario al unir los componentes definidos por el usuario y los propios componentes que pueden ser desplegados en el marco de la aplicación en tiempo de ejecución.

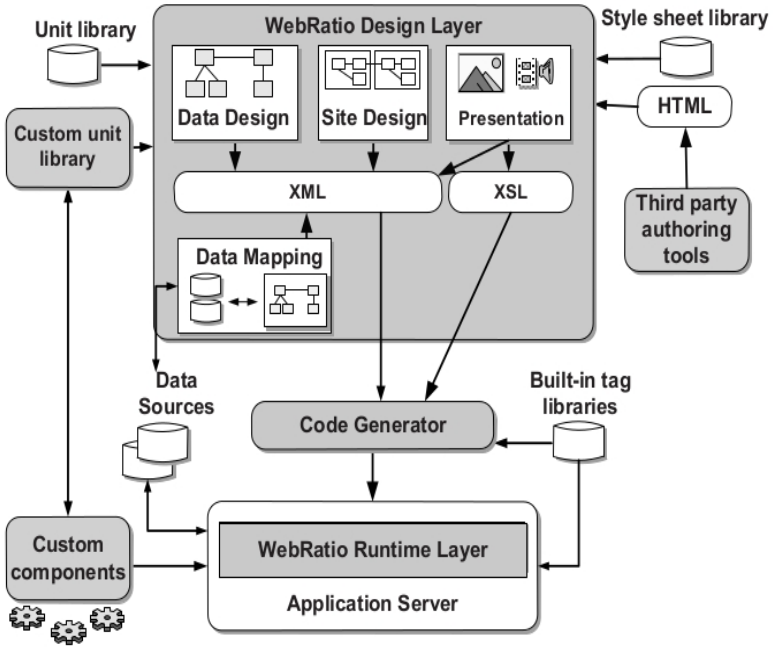


*Figura 3-7. Arquitectura WebRatio. Fuente: Brambilla, 2012*

## *3.8. Marco de referencia*

La solución propuesta en esta investigación para la implementación tiene base en el enfoque Model-Driven, que se resume en el esquema de la Figura 3-8. En esta figura se observa la propuesta de obtener un modelo que represente a los recursos a partir de un DSL gráfico, en cual debe ir transformándose en otros modelos hasta llegar a una base de datos relacional, en la cual se encontrará los diferentes recursos que representan el RI. Luego, a partir de esa base de datos se desarrollaran funcionalidades para el RI a través de WebRatio, gracias a los modelos que exige dicha herramienta.

El ideal para los actores involucrados (desarrolladores, dueños del negocio y expertos del dominio) en el diseño y desarrollo de un sistema para repositorios, es que exista un lenguaje neutral y de alto nivel que permita que las partes se pongan de acuerdo y que les sirva de apoyo para describir, discutir y negociar las funciones (destinados a recopilar, catalogar, almacenar, gestionar, acceder, difundir y preservar)

que el repositorio debe ofrecer. El enfoque *Model-Driven* brinda un marco que permite a los interesados compartir sus puntos de vista y manipular directamente las representaciones de las entidades de este dominio. Además, este paradigma ofrece algunas ventajas como: incremento en la productividad (errores, costos, código), adaptación a cambios tecnológicos, reuso de software, mejora en la comunicación con usuarios y desarrolladores, asignación de roles, entre otros (Pons et al., 2010). Por ello, esta investigación implementó MDE en varias fases o módulos, que son ampliados en los siguientes capítulos.

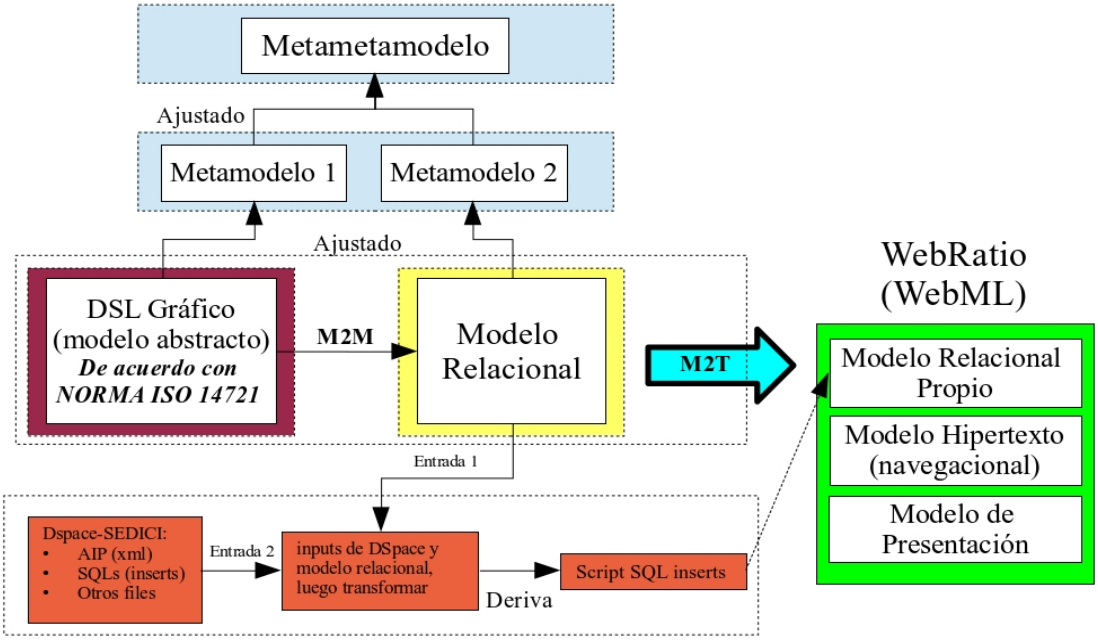


*Figura 3-8. Esquema general de la propuesta*

# CAPÍTULO 4
# CASOS DE ESTUDIO

La concepción que se tiene de las bibliotecas tradicionales es ofrecer documentos (libros, revistas, tesis, artículos, etc.) en soportes físicos a través de servicios de préstamos y consultas. Todo esto ha evolucionado hasta converger, hoy en día, en las bibliotecas digitales y los repositorios institucionales, con una diversidad de áreas, que se estudian e investigan para ofrecer mejores y nuevos servicios a los usuarios. En los capítulos 2 y 3, se mencionó el marco teórico que guía esta investigación destinada a mejorar la representación de recursos en un repositorio institucional. Esto sirve de base para exponer varios casos prácticos que se han estudiado e investigado a lo largo de toda la investigación doctoral y muestran la necesidad de hacer uso de la propuesta desarrollada.

En este capítulo se realiza una revisión bibliográfica de las diferentes líneas de investigación que relacionan el enfoque Model-Driven con el dominio de los RI y/o BD, conocido como LIS – Library & Information Science – (Chua & Yang, 2008; McNicol, 2003; Nguyen & Chowdhury, 2011; Yang et al., 2006), luego se analiza una propuesta de desarrollo de aplicaciones web para el dominio de estudio a partir del Model-Driven y, finalmente, se muestran diferentes alternativas de incorporación de recursos a SEDICI, situación que expone la necesidad de contar con una representación de recursos para solventar esos problemas que se exponen en esta sección.

## *4.1. Propuestas existentes bajo Model-Driven*

En la publicación de Texier et al. (2014) se analizaron trabajos que señalaban una relación entre el enfoque Model-Driven y el mundo LIS a partir de bases de datos de artículos y conferencias referentes del área. Se tomaron en cuenta 5 artículos resumidos en la línea de investigación de Malizia *et al*. (Guerra, de Lara, Malizia, & Díaz, 2009;

Guerra, Lara, & Malizia, 2007; Malizia et al., 2010; Malizia, Bottoni, Levialdi, & Astorga-Paliza, 2007), y la investigación de Paganelli & Pettenati (2006) que se sintetizan.

En el trabajo de Paganelli & Pettenati (2006) se explica cómo los enfoques dirigidos por modelos pueden ayudar a los sistemas de gestión de documentos basados en modelos de información en XML. También describe las características de los documentos, los requerimientos para estos sistemas y un framework de desarrollo.

En el trabajo de Malizia et al. (2010) describe un framework basado en un metamodelo y un lenguaje visual. Este framework, llamado por los autores CRADLE (Cooperative-Relational Approach to Digital Library Environments), es una definición de conceptos y servicios relacionados con el desarrollo de bibliotecas digitales compuesto por cinco entidades: actor, colección, servicios, estructura y documento. Este artículo reseña los antecedentes del enfoque dirigido por modelos en las BD y describen la arquitectura y el metamodelo del framework. Las características principales del framework son: tener un lenguaje visual que ayuda a las partes interesadas integrando el modelado con la generación de código y en la definición de metadatos. Finalmente, estos autores presentan cómo es la generación de sistemas en este dominio y su evaluación.

Estos resultados reflejan un área de vacancia entre los modelos conceptuales para repositorios digitales y el desarrollo de software dirigido por modelos, es decir, el Model-Driven no ha sido determinante en el diseño e implementación de sistemas de repositorios, ya sea por ser un paradigma nuevo o por no haber sido (aún) adoptado por la comunidad de informáticos, menos aún en la representación de recursos.

## *4.2. WebRatio en los RI*

Un trabajo técnico realizado por el personal de SEDICI (Texier, De Giusti, Lira, & Villarreal, 2014), relevó que los diferentes RI se desarrollan a través de plataformas de software que permiten gestionar los diferentes servicios que prestan Algunas instituciones utilizan plataformas preexistentes comunes como DSpace, EPrints y Digital Commons (OpenDOAR, 2014; ROAR, 2014), mientras que otras realizan

desarrollos propios, sobre las cuales se gestiona un gran porcentaje de los recursos depositados en el mundo, a saber: arXiv, CiteSeerX, Social Science Research Network, entre otras (CSIC, 2014).

Todas estas plataformas de software deben ser diseñadas e implementadas a través de alguna metodología de desarrollo de software. En este contexto, existen los factores que determinan el diseño e implementación de componentes de software en un repositorio para buscar soluciones a sus problemas. Estos factores incluyen: la diversidad de soluciones tecnológicas, tratamiento de diferentes tipologías de recursos, esquemas de metadatos, recomendaciones de almacenamientos de recursos, la preservación de recursos, recomendaciones de modelos conceptuales o de datos como: OAIS, FRBR, modelo 5S, modelo DELOS, Europeana Data Model, etc.

El objetivo del estudio técnico fue visualizar y modificar los autores de los recursos de tipo "Tesis" en el repositorio institucional de la UNLP (SEDICI), para dar una solución que permitiera abordar el problema en los RI sin tener que comprender la plataforma de software, más allá del conocimiento (o desconocimiento), esto permite a los distintos actores en este sistema de software (desarrolladores, dueños del negocio y expertos del dominio) proponer soluciones e ideas independientes de la plataforma subyacente.

Los recursos que se describen a partir de la enumeración de los metadatos son representados a través de distintos esquemas (formatos), que si bien comparten una sintaxis y estructura de la información, difieren atendiendo a los propósitos de la información que describen (Méndez, 2003b). La entidad abstracta *autor*, en SEDICI, es el metadato de los recursos que permite identificar el creador intelectual del recurso, ya sea una persona o una organización. El *autor* se considera entidad abstracta por poseer información descriptiva propia, utilizada en los procesos de catalogación de recursos como elemento de un vocabulario controlado (Texier, De Giusti, Oviedo, et al., 2013). Ante esto, se entiende la importancia de poder localizar el *autor* de un recurso para identificarlo, a través de búsquedas o navegación en la plataforma de software.

Los *autores* son catalogados de diferentes maneras en las distintas implementaciones generadas a partir de diversas plataformas de software de repositorios, ya que existen múltiples formas de normalizar dichos autores para el

funcionamiento e interoperabilidad de los RI. Por ello, es necesario tener un control completo de la entidad *autores* de manera que cualquier plataforma de software pueda gestionar la normalización de la forma que desee, para esto, es recomendable que un *autor* tenga un registro lo más completo posible, es decir, entre más datos y nombres completos se tenga, será mejor para la gestión del RI. Por ello, el estudio realizado en SEDICI, dio cuenta de WebRatio como una herramienta para gestionar dicha entidad abstracta siguiendo una metodología de desarrollo de software dirigida por modelos. En la Figura 4-1 se observa cómo es el proceso WebRatio en menor escala, es decir, en la fase de modelado. Allí se puede sincronizar y modificar el esquema de la base de datos del repositorio así como también diseñar el modelo de hipertexto o navegacional, después se pueden realizar algunas personalizaciones, para generar la aplicación web en forma automática y, finalmente, usar la aplicación.

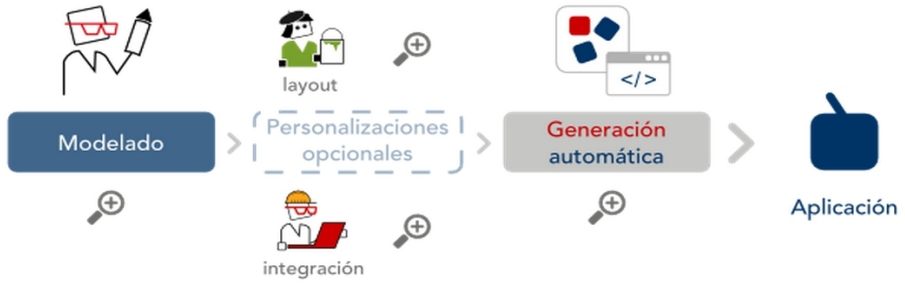


*Figura 4-1. Proceso WebRatio. Fuente: WebRatio.com*

WebRatio permitió visualizar y modificar los *autores* del Servicio de Difusión de la Creación Intelectual (SEDICI, 2013) y crear las funciones como el ABM (alta, baja y modificación) para dichos *autores*. SEDICI, usa la plataforma de software DSpace. Esta es una plataforma de código abierto con una arquitectura de tres capas (negocio, aplicación y almacenamiento) para la gestión de recursos digitales que se organizan a partir de comunidades jerárquicas y colecciones (DSpace, 2014). DSpace ofrece dos frontend, XMLUI y JSPUI, y deja abierta la gestión de la entidad *autores* en forma particular. SEDICI, decidió gestionar la entidad *autores* de la misma forma como la realizaba en la plataforma de software de la que migró en el 2012 (Celsius DL) y que está basada en una base de datos MySQL. Entonces, WebRatio, brinda la posibilidad de gestionar la relación de DSpace-SEDICI con la base de datos de autores en MySQL de

forma distinta. Por tanto, el estudio técnico permitió concluir lo siguiente:

- Se cumplió con el propósito de poder visualizar y modificar la entidad *autores* en SEDICI a través del enfoque Model-Driven, es decir, un problema en concreto solucionado con una determinada tecnología. Este enfoque ayuda a que las partes involucradas en un sistema de repositorios hablaran un mismo idioma sin importar la plataforma de software de RI usada.
- Se observó el potencial que puede tener WebRatio en los repositorios institucionales, ya que se pueden implementar diversas funcionalidades sin importar la arquitectura de software en la que se encuentra el repositorio.
- La propuesta del estudio se enfocó en la plataforma de software DSpace en SEDICI como campo de pruebas, pero esta propuesta puede aplicarse a otros RI con otras plataformas de software.

## *4.3. Incorporación de recursos en SEDICI*

### 4.3.1. Celsius DL

En sección 1.1 del capítulo 1 se hizo referencia, en forma general, al trabajo que remite a las dificultades presentadas con Celsius DL en el 2011 (De Giusti et al., 2011), que contaba para ese año con 14.000 recursos académicos propios de la UNLP expuestos bajo políticas de acceso abierto. Los siguientes problemas se presentaron con la plataforma de software:

- Tareas de mantenimiento diario al software de soporte, ya que se desarrolló una solución de software nueva y a la medida.
- Aporte de muchos desarrolladores con códigos fuentes.
- La necesidad de mejorar servicios existentes y agregar nuevos servicios.
- Adaptación a los avances tecnológicos.
- Desarrollos de aplicaciones en código abierto en el dominio LIS, para una mayor colaboración por parte de la comunidad.
- Obsolescencia tecnológica que se evita.
- Representación de recursos, catalogación de recursos, recuperación de

información, diseminación selectiva de información, autoarchivo, entre otros, presentan inconvenientes porque son independientes a las implementaciones que se realizan con diversas plataformas de software.

Los problemas generaron la necesidad a SEDICI de analizar y pensar en un cambio de plataforma de gestión del repositorio tomando en cuenta que el nuevo software reuniera las siguientes características: licencia de uso abierta, administración, personalización, documentación, actualizaciones, soporte, escalabilidad, interoperabilidad, entre otras. Por ello, la principal dificultad que se presentó fue la representación de recursos y en cómo importar todos los recursos de Celsius DL al nuevo SEDICI (en DSpace). Desde luego, de haber contado con una herramienta que centralizara la representación de los recursos, independientemente, de la plataforma usada, la importación no hubiese ocasionado tantos problemas y no se hubiesen invertido tantos recursos cuantificado en las horas hombre usadas para atender esta problemática.

### 4.3.2. Portal de congresos y revistas

En el 2013, el personal de SEDICI presentó una propuesta para facilitar la interacción entre los portales web de gestión congresos y revistas de la UNLP –con los paquetes de software de PKP, OCS y OJS respectivamente– y el repositorio institucional de la UNLP (De Giusti et al., 2013), con el objetivo de maximizar la difusión abierta del conocimiento y evitar la duplicidad de esfuerzos.

Los mecanismos que se analizaron fueron la utilización de diversos protocolos como RSS/Atom, SWORD y OAI-PMH, y el desarrollo de nuevos plugins para permitir la interrelación con dichos portales. Estos caminos tecnológicos deben estar relacionados con el establecimiento de flujos de trabajo que permitan, a los servicios desarrollados, una colaboración de ida y vuelta con SEDICI para asegurar la preservación de la producción realizada y minimizar el re-trabajo por parte de los integrantes de ambos servicios. Por ello, se presentaron las alternativas:

- De OJS a DSpace a través de exportaciones en XML y usando SWORD v1.;

- De DSpace a OJS mediante la utilización de canales o feeds RSS/Atom, y a través de protocolos OAI-PMH, SWORD y exportaciones de paquetes AIP desde DSpace;
- De OCS a DSpace gracias a un plugin desarrollado por SEDICI y por medio de SWORD2;
- Comunicación desde las diferentes facultades de la UNLP a SEDICI, por medio de los protocolos OAI-PMH y OpenSearch.

Con este trabajo se observaron los diferentes esfuerzos que se hacen actualmente en SEDICI y nuevamente, el contar con un marco de referencia para la representación de recursos como el que se plantea en esta investigación, permitirá la adaptación al repositorio SEDICI de los portales y repositorios que se encuentran en algunas facultades de la UNLP, ya que la información almacenada y adaptada a la plataforma de software del repositorio se extrae de acuerdo con el marco de referencia y luego se pudiera exportar a cualquier otro repositorio.

### 4.3.3. Scopus

A mediados del año 2013, se comenzó un estudio sobre la incorporación de los recursos de la base de datos bibliográfica Scopus a SEDICI, con filiaciones de la UNLP (Texier, Giusti, Lira, & Villarreal, 2014). El objetivo del estudio fue recuperar y dar cuenta de todas las obras producidas por autores de la UNLP, para dar difusión y visibilidad dentro de su repositorio institucional e incrementar las producciones presentes en el mismo. La recopilación fue de los artículos, objetos de conferencia y artículos de revisión comprendidos entre 1927 y 2013.

Para lograr tal objetivo, se definió un mecanismo semiautomático de incorporación de recursos desde Scopus a SEDICI que sirve de modelo a otras universidades e instituciones. Este mecanismo fue desarrollado en sucesivas fases, que pueden resumirse en: recuperación desde Scopus, filtrado, limpieza, transformación y normalización de los metadatos, importación de autores y de los recursos, revisión, filtrado, incorporación de los PDF (si la licencia lo permite) y finalmente publicación en el repositorio. La consulta sobre Scopus finalizó el 1 de marzo de 2014 en la UNLP .

Los metadatos exportados desde Scopus (formato CSV) se incorporaron a Google Refine (OpenRefine, 2014), lo que permitió realizar los análisis, transformaciones y normalizaciones de manera flexible y rápida. Luego se obtuvieron los listados de revistas de DOAJ, PubMed, SherpaRomeo, SciELO y RedALyC, para relacionar la información de esas revistas con todos los documentos importados a través del ISSN como identificador o clave única, agrupándose en las siguientes categorías: licencia Creative Commons, contenidos en DOAJ, acceso abierto pero sin licencia y contenido DOAJ, artículos en acceso abierto pero posiblemente en preprint o postprint, y, otros. A continuación, los documentos de esas categorías se exportaron desde Google Refine (archivos XML) para ser importados en DSpace vía formato Simple Archive Format. Con los documentos incorporados (aproximadamente 19.000) en el servidor de desarrollo de SEDICI, se iniciaron los procesos de eliminación de duplicados, mejora de la calidad de los metadatos y adquisición de PDFs. Esta información permitió sumar recursos a la producción local alojada en el repositorio, mejorar la calidad de los metadatos ya almacenados e incrementar la visibilidad del repositorio y de la universidad.

Este estudio hubiera podido simplificarse si se contara con un marco que permitiera la representación de dichos recursos (que provienen de Scopus) de forma independiente, para luego adaptarlos a SEDICI o a cualquier otro repositorio.

En conclusión, presentar estos casos de estudio dan cuenta de los distintos caminos que hay que transitar para realizar incorporaciones o mejorar la calidad de los metadatos de los recursos de SEDICI, los cuales se obtienen a partir de fuentes muy diversas (bajo DSpace), situación que lo haría más abstracto y más amplio si las importaciones también estuvieran diseñadas desde otras plataformas de repositorios, por ejemplo, EPrints o Fedora, y para diferentes implementaciones y personalizaciones sobre estas plataformas, las cuales varían considerablemente entre las diferentes instituciones donde se implementan.

# CAPÍTULO 5
# METODOLOGÍA PROPUESTA PARA LA REPRESENTACIÓN DE RECURSOS

Los casos de estudios descritos en el capítulo anterior explican el problema planteado en esta investigación y dan cuenta de la necesidad de una solución para la representación de recursos dentro del contexto de los Repositorios Institucionales (RI). Se ha mencionado que la representación de recursos es un problema recurrente (Candela et al., 2007; Fox et al., 2012; Gonçalves et al., 2004; Malizia et al., 2010; Paganelli & Pettenati, 2006), entre otros. Y los trabajos abordan el tema como sistemas de repositorios diseñados en forma general  a diferencia de ellos, este trabajo introduce el recurso como el eje central.

En este capítulo se da cuenta del marco de referencia para definir un modelo que permita el desarrollo de funcionalidades en los RI a partir de la representación de recursos, y se relaciona con los siguientes elementos de los repositorios: recursos, esquemas de metadatos, almacenamiento y catalogación. A su vez la propuesta se basa en los procesos funcionales relacionados con el material depositado y la norma ISO 14721: carga (ingest), almacenamiento (storage), catalogación (cataloging), indización (indexing), búsqueda (search engine) y navegación (browsing). Para lograrlo, se hizo un relevamiento bibliográfico de los modelos usados en este dominio y se analizaron diferentes plataformas de repositorios, necesarios para implementar la propuesta (Texier & De Giusti, 2014).

De igual manera, en este capítulo se desarrollo la propuesta del marco de referencia que se estructuró en 5 módulos o fases (Figura 5-1) y presenta un análisis del contexto del problema planteado con un diseño, una implementación y una descripción de los beneficios en cada una de las 5 fases, a saber:

1. DSL para el desarrollo del modelo flexible.

2. Transformación del modelo diseñado con el DSL a un modelo relacional.
3. Transformación del modelo relacional a un script SQL para la crear la base de datos.
4. Mapping de los recursos de Dspace-SEDICI con la base de datos creada.
5. Desarrollo de una aplicación en WebRatio para la visualización y exportación de los recursos que se encuentran en la base de datos obtenida.

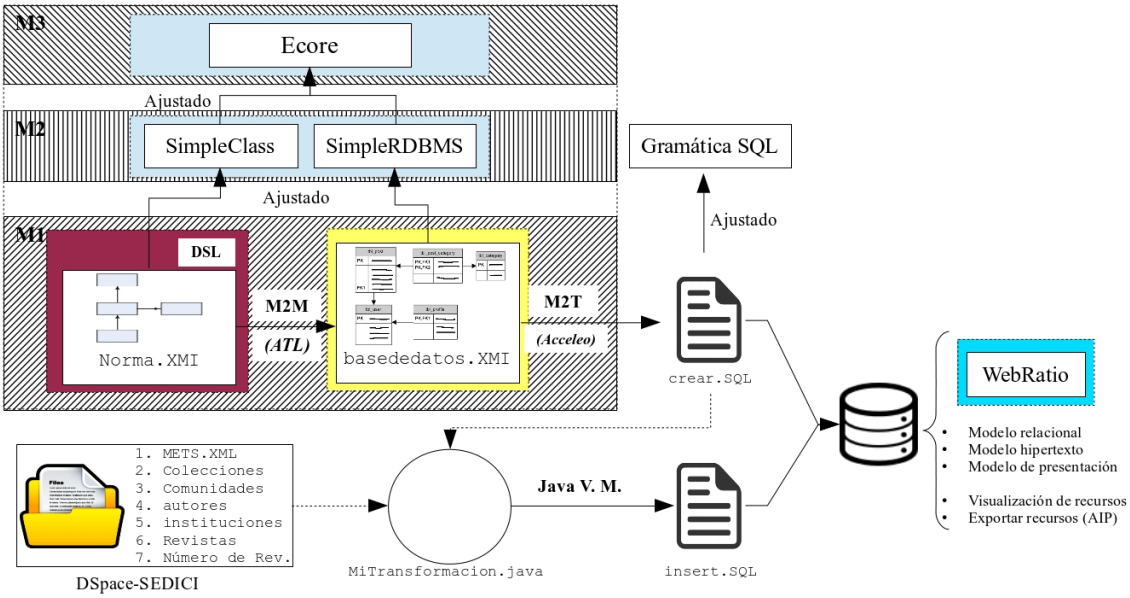


*Figura 5-1. Marco de referencia propuesto*

Para el desarrollo de las fases se seleccionaron un conjunto de herramientas bajo el enfoque Model-Driven, pensadas para automatizar el proceso de desarrollo de software. A diferencia de las herramientas de modelado como UML que nacen con el objetivo principal de permitir la especificación de sistemas de software, es decir, generar código, generar documentación, etc. Por ello, UML no es un lenguaje específico del dominio, sino concretamente un lenguaje de modelado general y permite extensiones para adaptarlo a un dominio específico mediante el uso de perfiles.

Adicionalmente, las herramientas usadas en esta investigación remiten a la filosofía Free and Open Source Software (FOSS) manteniendo los criterios legales que establecen tales licencias (FOSS, 2014). Las razones de esta selección, se enmarcaron en el contexto de la libertad e independencia en el manejo de los paquetes de software, permitir futuras mejoras y colaboraciones a la propuesta, y a la filosofía de trabajo de

colaboración del personal de SEDICI. Por ello, el entorno de desarrollo de software que se adapta a las necesidades deseadas es Eclipse (Eclipse, 2014a) con el componente de software Eclipse Modeling Project, desde ahora EMP (Gronback, 2009). Se destaca que se estableció un control de versiones del código fuente del software generado durante la investigación para cada fase a través del proyecto GitHub (GitHub, 2014a).

## *5.1. Análisis del contexto*

### 5.1.1. Elementos de la representación de recursos

En una representación de recursos dentro de los repositorios institucionales se definen cuatro elementos (CCSDS, 2012; Texier, De Giusti, Oviedo, et al., 2013), presentes dentro del marco de referencia propuesto:

- tipología de los recursos,
- esquemas de metadatos,
- almacenamiento, y,
- catalogación representada por los vocabularios controlados, tesauros y las entidades abstractas tales como autores, instituciones y revistas.

#### *5.1.1.1. Tipologías de recursos*

Los recursos son generados por diferentes razones. En esta tesis se remite al subconjunto de tipologías que predominan en SEDICI, a sabiendas de que la diversidad de recursos tienden a aumentar con el paso del tiempo de acuerdo con las necesidades de los usuarios y la evolución de las tecnologías. Los recursos son:

1. Artículos de investigación. Documento que expresa, por parte del autor, una investigación redactada en forma especializada. Por lo general, tienen un aporte a la comunidad científica que debe incluir referencias que permitan verificar y reproducir los aportes que se dan a conocer. La verificación es conocida como revisión por pares, por ello, siempre son publicados en revistas arbitradas.
2. Tesis. Es un trabajo científico sometido a un sistema de reglas que dependen del grado académico al que se esté optando, por ejemplo: grado, especialización, maestría, doctorado. Los trabajos se clasifican en: tesina de grado, trabajo de

especialización, tesis de maestría y tesis de doctorado.

3. Libro. Es una obra científica, literaria o de cualquier otra índole impresa o en cualquier otro soporte. Actualmente los libros pueden ser escaneados o producidos en algún formato electrónico.
4. Entidades abstractas. Conjunto de elementos que poseen información descriptiva propia. En esta propuesta se trabaja con tres tipos de entidades:
    a) Entidad abstracta *autor*. Representa la información del creador de algún tipo de recurso, del director/tutor de una tesis, del jurado, etc.
    b) Entidad abstracta *institución*. Identifica la entidad a la que pertenece el autor y/o el recurso.
    c) Entidad abstracta *revista*. Es una publicación periódica que agrupa un conjunto de artículos científicos de diversas disciplinas y asegura su calidad a través de la revisión por pares (arbitraje).

#### *5.1.1.2. Esquema de metadatos*

Los trabajos de Méndez (2003b) y Riley (2010) dan cuenta de la cantidad de esquemas de metadatos (conocidos también como formatos o estándares de metadatos) que se pueden usar para describir un recurso, y que pueden encontrarse en diversas implementaciones de repositorios con propósitos de información distintos. En este contexto, se generan problemas con los esquemas que deben resolverse a partir de la clasificación de los mismos, ya sea por su estructura o por su especificidad.

Según su estructura pueden ser planos (no existe anidamientos de metadatos) y jerárquicos, donde sí existe anidamientos de metadatos. Con respecto a la especificidad pueden ser simples (pocos elementos, y por ende, más generales) o complejos (ofrecen mayor riqueza estructural y semántica, por lo tanto, permiten ser más específicos en la catalogación del material). Los problemas se complejizan en formatos jerárquicos porque si se está usando una base de datos relacional, las consultas SQL serían muy complejas degradando su performance. Por ello, una opción viable es el uso de formatos anidados como XML con una base de datos de ese tipo. Por el contrario, si se usan formatos simples, su tratamiento estaría solamente limitado a elementos con un nombre (clave) y un valor.

Por ello, una solución para los problemas presentados con los formatos en el contexto de esta indagación (DSpace no soporta esquemas de metadatos jerárquicos), es limitarse a los esquemas existentes o a estructuras propias no anidadas. En la actualidad, DSpace-SEDICI cuenta con un esquema propio y combinado con dos esquemas conocidos como DC y una emulación de MODS (se usa el nombre del campo como elemento y el nombre del campo hijo como calificador). Adicionalmente, se conoce que DSpace sólo puede tratar con formatos de metadatos planos y una forma de "simular" esta jerarquización es estableciendo una relación entre metadatos de detalle (o hijo) y un metadato principal (o padre). Por ejemplo, los metadatos <autor> y <filiacion> en un formato plano no tienen relación entre sí, por lo que no hay certeza sobre qué filiación corresponde a cada autor.

```
<autor>Oviedo, Nestor F.</autor>
<autor>Texier, Jose</autor>
<filiacion>Universidad Nacional de La Plata (UNLP)</filiacion>
<filiacion>Universidad del Táchira (UNET)</filiacion>
```

En el caso de contar con un formato de metadatos jerárquicos, estas ambigüedades no existirían. Por tanto, la estructura anidada sería:

```
<autor>
      <nombre>Oviedo Nestor F.</nombre>
      <filiacion>Universidad Nacional de La Plata (UNLP)</filiacion>
</autor>
<autor>
      <nombre>Texier, Jose</nombre>
      <filiacion>Universidad del Táchira (UNET)</filiacion>
</autor>
```

En este ejemplo, si se adaptara a DSpace, el metadato principal sería <autor>, mientras que el metadato de detalle sería <filiacion>. Considerando que existe un ID único para cada metadato, esto quedaría de la siguiente manera:

```
<autor id="1">Oviedo Nestor F.</autor>
<autor id="2">Texier, Jose</autor>
<filiacion id="3" padre="1">Universidad de La Plata (UNLP)</filiacion>
<filiacion id="4" padre="2">Universidad del Táchira (UNET)</filiacion>
```

Por ello, en SEDICI aún no se ha definido la utilidad de tener campos hijos en más de un nivel. La simplicidad o no de permitir estas relaciones estará dada en parte por las alternativas de implementación que se planteen.

A pesar de esto, el estudio se centró en cierta tipología de recursos (sección 1.3) entendiendo que esta con el paso del tiempo tiende a aumentar de acuerdo con las necesidades de los usuarios y la evolución de las tecnologías. Los metadatos que se eligieron para esta investigación en cada uno de los recursos relevados fueron:

1. Artículos de investigación:
    a) Título.
    b) Subtítulo.
    c) Fecha de publicación.
    d) Idioma.
    e) Entidad abstracta autor*[1].
    f) Entidad abstracta institución*.
    g) Descripción física.
    h) Palabras clave.
    i) Descriptores.
    j) Tipo de documento.
    k) Identificadores geográficos.
    l) Abstract.
    m) URL de acceso.
    n) Nota.
2. Tesis:
    a) Título del documento.
    b) Subtítulo.
    c) Entidad abstracta autor*.
    d) Entidad abstracta institución.
    e) Fecha de presentación.
    f) Director de la tesis (entidad abstracta autor).
    g) Co-director de la tesis* (entidad abstracta autor).
    h) Miembros del jurado* (entidad abstracta autor).
    i) Descriptores.
    j) Descripción del recurso.
    k) Grado alcanzado.
    l) Palabras clave.
    m) Lugar de desarrollo.
    n) Idioma.

[1] * significa de aquí en adelante, que ese metadato puede repetirse

o) Tipo de documento.
p) Abstract.
q) Localización física.
r) Localización electrónica.
s) Nota.

3. Libro:
   a) Título del documento.
   b) Subtítulo.
   c) Entidad abstracta autor*.
   d) Fecha de publicación.
   e) Idioma.
   f) Editor*.
   g) ISBN.
   h) Editorial.
   i) Descripción física.
   j) Palabras clave.
   k) Descriptores.
   l) Tipo de documento.
   m) Localización electrónica.
   n) Nota.
4. Entidad abstracta autor:
   a) Nombres.
   b) Apellidos.
   c) Correo electrónico*.
   d) Ciudad.
   e) País.
   f) Entidad abstracta institución*.
   g) Miembro de grupos.
   h) Fecha de nacimiento.
   i) Fecha de defunción.
   j) Profesión*.

k) Ocupación.
l) Sexo.
m) Idiomas.
n) Alias*.
o) Nota.

5. Entidad abstracta institución:
   a) Nombre.
   b) Ciudad.
   c) País.
   d) Dirección.
   e) Alias*.
   f) Nota.
   g) Dependencias*.
6. Entidad abstracta revista:
   a) Título de la serie.
   b) Subtítulo.
   c) ISSN.
   d) Frecuencia.
   e) Editor*.
   f) Materias*.
   g) Indizada*.
   h) Director* (entidad abstracta autor).
   i) Subdirector* (entidad abstracta autor).
   j) Comité de referato* (entidad abstracta autor).
   k) Secretaría de redacción* (entidad abstracta autor).
   l) Volumen.
   m) Número.
   n) Tipo de documento.
   o) Localización física.

*5.1.1.3. Almacenamiento*

Todos los metadatos de los recursos de un repositorio institucional deben almacenarse de forma persistente. Este almacenamiento debe ser configurable a cualquier tipo de paradigma de base de datos e independiente del modelo de representación de los recursos definidos. Para ello, siempre es necesario analizar alternativas de acuerdo con la plataforma de software usada, desde el punto de vista de la performance (tiempos de respuestas y consumo de recursos), flexibilidad y escalabilidad (mantener sus cualidades aún aumentando sus recursos, usuarios, etc). Algunas opciones para la persistencia de los datos pueden ser en XML (p.e. eXist), relacionales (p.e. PostgreSQL), orientadas a objetos (p.e. Versant), RDF (p.e. Apache Jena), o soluciones mixtas.

En DSpace-SEDICI la persistencia de los datos es asegurada bajo un motor de base de datos relacional conocido como PostgreSQL 8.1. Para garantizar la simulación de un esquema de metadatos jerárquicos, se deben realizar pequeños ajustes en tablas particulares de la base de datos de DSpace, detalle que siempre compromete la migración del DSpace para otras versiones, ya que son cambios no propios de DSpace sino una personalización para SEDICI. Por ejemplo, la tabla *metadatavalue* es donde se almacenan los metadatos, y tiene los siguientes campos:

- metadata_value_id: id del metadato.
- item_id: id del recurso.
- metadata_field_id: id del tipo de metadato (autor, title, etc).
- text_value: valor textual del metadato.
- text_lang: idioma del valor textual.
- authority: id del elemento correspondiente a un vocabulario controlado.

Para llevar a cabo esta pseudojerarquía, sería necesario agregar un campo a esta tabla (por ejemplo parent_metadata_value_id), en el cual se indique el metadato de nivel superior en la jerarquía.

*5.1.1.4. Catalogación*

Los procesos de catalogación de recursos se realizan en dos fases (Martínez-Tamayo &

Valdez, 2009): análisis documental formal (ADF) y análisis documental de contenido (ADC). El ADF se entiende como la descripción física/bibliográfica o forma externa del documento, es decir, los aspectos formales del recurso a través de los metadatos descriptivos y está relacionada con el uso de normas de catalogación como las Resource Description and Access (RDA), Anglo-American Cataloguing Rules (AACR) o International Standard Bibliographic Description (ISBD).

El ADC es conocido como la indización del recurso y describe el contenido intelectual (textual-temático), el cual está definido en dos niveles, lenguaje natural o contenido textual donde se completan el título, palabras clave, resumen y en algunos casos el texto completo; y, el otro nivel, lenguaje documental de control o contenido temático a través de tesauros (Eurovoc o DeCS), listas de encabezados de materias (LEMB, SEARS, LCSH) y sistemas de clasificación del conocimiento (CDU, CDD, LCC). En síntesis, muchos campos presentes en los esquemas de metadatos son estandarizados y su identificación, recuperación y uso se hacen de acuerdo con las fases nombradas. Para evitar la redundancia y garantizar la integridad de la información en los vocabularios controlados, se pueden considerar tres puntos de vista en la catalogación (De Giusti et al., 2011):

- Forma de representación: puede estar en una lista simple de elementos o en elementos relacionados; se encuentran en una tabla de una base de datos, en un archivo XML con un esquema particular, en un archivo de texto, etc.
- Forma de referenciar: se necesita una relación para distinguir de manera unívoca un elemento en un vocabulario determinado, por ello, se debe tomar la decisión entre: un metadato vacío con un valor adicional para la referencia, un metadato con valor del vocabulario replicado junto con un dato adicional para la referencia y un metadato con la referencia como valor.
- Forma de presentación: es como el usuario observará y utilizará los metadatos en el portal, es decir, en formularios de carga, en la página de presentación de metadatos, en la exportación de recursos, etc. La forma puede ser simple, intuitiva, e internacionalizable.

Un punto aparte merecen los problemas de catalogación presentados en las entidades abstractas en los procesos de catalogación (M. De Giusti et al., 2011).

Algunos ejemplos de entidades abstractas son los autores, instituciones, eventos, revistas y sus números. Se pueden considerar los mismos puntos de vista que en los vocabularios controlados (representación, referencia y presentación):

- Forma de representación: depende del esquema de metadatos seleccionado, similar a los mismos problemas descritos anteriormente en los vocabularios controlados. Adicionalmente, se puede pensar en el uso de servicios web como proveedor de entidades.
- Forma de referenciar: una vez seleccionada una entidad abstracta es necesario guardar la referencia, lo cual trae problemas de compatibilidad entre la representación elegida para la entidad abstracta y los metadatos del recurso a los cuales se asocia esa entidad.
- Forma de presentación: además de las consideraciones presentes en los vocabularios controlados, es necesario considerar los problemas generados en el formato de catalogación usado. Se tienen dos alternativas: en el momento de catalogación debe realizarse una transformación única que elimine el problema de duplicidad y de consistencia; y en el momento de presentación se requiere de una transformación cada vez que se muestre el recurso. Esto implica una mayor carga de procesamiento, que evita la duplicidad y asegura consistencia.

En SEDICI la necesidad de normalizar algunos campos en los esquemas de metadatos para que los recursos pueden ser identificados, recuperados y usados de forma clara, pone en contexto los vocabularios controlados. En otras palabras, estos vocabularios son usados para representar, referenciar y presentar los valores de los campos de una forma normalizada para los usuarios y administradores del sistema (De Giusti et al., 2011).

Durante algunos años los recursos se catalogaron temáticamente mediante el uso de descriptores (términos controlados) y palabras-clave (términos no controlados). Por descriptores se entiende un vocabulario finito y controlado de términos mientras que las palabras-clave surgen de los propios textos, proporcionadas por sus autores. En versiones anteriores de la plataforma usada en SEDICI (Celsius DL), se había implementado el uso de un listado de términos controlados al que se denominó “Materias”, en el cual se incluyó un conjunto restringido de términos controlados,

seleccionados por administradores idóneos, que hacen referencia a las grandes áreas temáticas del conocimiento. De este modo, los recursos son catalogados en primer término mediante una "macrocatalogación" (materias), luego una catalogación temática más restringida (descriptores) y finalmente, si las hay, mediante las palabras-clave proporcionadas en el texto por su autor. Puede decirse, sintéticamente, que se parte de lo general para llegar a lo particular de cada recurso.

En el mismo sentido, se hace necesario mencionar el tema de los descriptores, los cuales están contenidos en una estructura jerárquica que establece las relaciones entre ellos, denominada tesauro. Existe gran cantidad y variedad de tesauros. Durante mucho tiempo, Celsius DL utilizó el tesauro de la UNESCO y un tesauro propio, elaborado con base en el tesauro de la UNESCO, que incorporaba términos que no se encontraban allí y que eran necesarios para los recursos existentes en el repositorio. Con el crecimiento del repositorio fue evidente que dichos tesauros no alcanzaban a cubrir todas las necesidades de los recursos y, tras un adecuado estudio de los tesauros disponibles, se decidió incorporar dos nuevos tesauros: el Eurovoc y el DECs. Este último, a pesar de estar orientado mayormente hacia las ciencias de la salud incluye también numerosos términos de otras áreas, con lo cual su inclusión ha resultado más beneficiosa de lo que se esperaba. Por ejemplo, en Celsius DL si se quería identificar la temática de un libro sobre "*Edificios e instalaciones oficiales de enseñanza media*" según el lenguaje documental usado, sería de la siguiente manera (Tabla 5-1):

| Lenguaje Documental Usado | Traducción al Lenguaje Documental |
|---|---|
| Sistema de clasificación decimal | 371.6 |
| Lista de encabezamientos de materias | Arquitectura escolar |
| Tesauro | Escuelas, espacios educativos, instalaciones, normas de construcción, diseño arquitectónico |
| Lista de términos libres | Escuelas-rancho |

*Tabla 5-1. Identificación de un libro*

Las entidades abstractas son formas de agrupar recursos que, por una u otra razón, deben presentarse visualmente juntos, como en el caso de los artículos de un

número determinado de una revista. Al igual que los vocabularios controlados, las entidades abstractas son usadas para representar, referenciar y presentar valores de metadatos en forma normalizada. Por ejemplo, para evitar que los artículos se presenten en forma desordenada o apartada es que se los incluye dentro de estas "entidades" que tienen la función de presentar la información ordenadamente. Sin embargo, esto supone un doble trabajo para los administradores: si, por caso, desean cargar un artículo de una revista, por un lado, deben generar una primera entidad abstracta, la Serie Documental, que comprende sólo los datos generales de la revista (nombre, director, frecuencia), y, por otro, una segunda entidad abstracta, la Entrega Documental, que hace referencia al número o volumen específico de la dicha revista donde fue publicado ese artículo. Sólo cuando estas dos primeras entidades están generadas, es posible que los administradores puedan cargar los artículos en cuestión.

DSpace-SEDICI cuenta con 30 tipos de documentos, entre los cuales se encuentran las entidades abstractas: serie documental o publicación periódica, entrega documental o número de publicación periódica, congresos y sus instancias (SEDICI, 2014). Además de las entidades abstractas presentes en las tipologías documentales, también existen las entidades abstractas *autores* e *instituciones*.

### 5.1.2. Modelos para repositorios institucionales

Los modelos son un conjunto de elementos que sirven para demostrar la consistencia de una teoría, es decir, representan con detalle un sistema dado (Brambilla et al., 2012; Pons et al., 2010). Los modelos de RI están enmarcados en los problemas originados por la representación de los recursos y la diversidad de soluciones tecnológicas disponibles en los distintos módulos funcionales: carga, almacenamiento, catalogación, indexación, búsqueda, navegación y preservación de los recursos. A continuación se describen los modelos conceptuales y modelos de datos usados en Bibliotecas Digitales y/o Repositorios Institucionales que si bien no se enfocan en el objeto de estudio de esta investigación (el **Recurso**), permiten comprender la diversidad de estándares y servicios que se proveen mediante dichos modelos. De manera tal que describirlos se hace obligatorio para comprender cómo el **Recurso** será visibilizado al interior de los modelos (Fox et al., 2012; Texier, De Giusti, & Gordillo, 2014).

*5.1.2.1 Modelos conceptuales para repositorios institucionales*

*Marco conceptual de Bawden y Rowlands*

Marco conceptual propuesto en 1999 por David Bawden y por Ian Rowlands. Se basa en una serie de términos esenciales para comprender el concepto de biblioteca digital y sus componentes (Bawden & Rowlands, 1999a). Estos autores se centran en tres aspectos cruciales orientados en la perspectiva del trabajo de Yates (1989), que pueden ser aplicados a las bibliotecas tradicionales y digitales: documentos, trabajo y tecnología. Luego, renombran cada uno de los tres aspectos mencionados: informacional (documentos), sistemas (tecnología) y social (trabajo).

- Dominio informacional: organización del conocimiento y su localización (p.e. Metadatos) e implicaciones sobre la cadena de transferencia de la información.
- *Dominio del sistema:* interacción persona-computador, organización del conocimiento para su fácil localización (p.e. agentes de software), impacto sobre la cadena de transferencia de la información.
- Dominio social: habilidades de información y la alfabetización, los impactos sobre las organizaciones y la naturaleza del trabajo, factores de gestión del RI, información legal y factores políticos, impacto sobre la cadena de transferencia de la información (técnica).

*El modelo formal 5S*

El modelo formal propuesto por Gonçalves et al. (2004), conocido como "Streams, structures, spaces, scenarios, societies" (5S). Fue propuesto sobre la base de cinco abstracciones las cuales ayudan a definir, relacionar y unir conceptos de objetos digitales, metadatos, colecciones y servicios, requeridos para formalizar las bibliotecas digitales.

- Streams (flujos) son secuencias de elementos de un tipo arbitrario (ej. bits, personajes, imágenes). Por tanto, se puede modelar el contenido estático y dinámico. Los streams estáticos corresponden a contenido de la información representada como elementos básicos, por ejemplo, un texto sencillo es una secuencia de caracteres, mientras que un objeto tan complejo como un libro puede ser una secuencia de texto simple y de imágenes. Los streams dinámicos

se utilizan para modelar cualquier flujo de información y, son importantes para la representación de cualquier comunicación que se lleva a cabo en la biblioteca digital.

- Structures (estructuras) son el camino a través del cual las partes de un todo están organizadas. En particular, se pueden utilizar para representar hipertextos y objetos estructurados de información, taxonomías, las conexiones del sistema y las relaciones de los usuarios.
- Spaces (espacios) son conjuntos de objetos junto con las operaciones sobre ellos, que se ajusten a ciertas restricciones. Los espacios de documentos son los conceptos claves en las bibliotecas digitales, sin embargo, los espacios son utilizados en diversos contextos y tipos de espacios.
- Scenarios (escenarios) son secuencias de eventos que pueden tener parámetros, en los cuales los eventos representan las transiciones de estado. Un escenario dice lo que ocurre con las streams en los espacios a través de las estructuras. Cuando se consideran en conjunto, los escenarios describen los servicios, las actividades y las tareas que representan las funciones de biblioteca digital.
- Societies (sociedades) son conjuntos de entidades y relaciones. Las entidades pueden ser seres humanos o los componentes de software y hardware, que utilizan o apoyan los servicios de bibliotecas digitales. Así, la sociedad representa el concepto de más alto nivel de una biblioteca digital, que existe para servir a las necesidades de información y para describir el contexto de su uso.

*Modelo DELOS*

El modelo general DELOS propuesto por Candela et al. (2007) consiste en un modelo para las bibliotecas digitales a partir del Manifiesto de DELOS dedicado a establecer las bases que fundamentan a las bibliotecas digitales a través de la identificación de sus conceptos. Este proyecto actualmente está coordinado por DL.org (2014). Establecen un framework llamado Three Tier Framework, con tres sistemas: Digital Library (DL), Digital Library System (DLS) y Digital Library Management System (DLMS). El primero, DL, es la organización que recoge, gestiona y preserva a largo plazo los contenidos digitales y que también los ofrece a las distintas comunidades de usuario. El

segundo, DLS es básicamente el sistema de software, la arquitectura requerida para brindar la funcionalidad de la biblioteca. Finalmente el DLMS es la plataforma en sí: sistema operativo, bases de datos, interfaz de usuario que brinda la funcionalidad básica y la integración con el software especializado. Mientras que el concepto de DL es abstracto, el DLS y el DLMS capturan realizaciones concretas de sistemas de software.

*El modelo conceptual FRBR*

El modelo conceptual FRBR (Functional Requirements for Bibliographic Records) fue propuesto en 1998 por la International Federation of Library Associations – IFLA – (FRBR, 2009). FRBR es una representación simplificada del universo bibliográfico. Está basado en un modelo entidad-relación para establecer un marco que proporcione una compresión clara, precisa y compartida por todos sobre la información de un registro bibliográfico. Este modelo conceptual consiste en identificar las entidades que son los principales objetos de interés para los usuarios de registros, las características o atributos de esas entidades y las relaciones entre entidades. FRBR (2009) forma tres grupos de entidades:

1. Grupo 1: representan los productos de creación intelectual o artística que se consignan o describen en los registros bibliográficos: obra (work), expresión (expression), manifestación (manifestation) e ítem (item). Constituyen el núcleo principal de las FRBR:
   - Obra: es una creación intelectual o artística diferenciada.
   - Expresión: la realización intelectual o artística de una obra en forma de notación alfanumérica, musical, o coreográfica, sonido, imagen, objeto, movimiento, etc., o cualquier combinación de dichas formas.
   - Manifestación: una expresión es la forma específica intelectual o artística que recibe una obra cada vez que se realiza.
   - Ítem: un ejemplar determinado de una manifestación.
2. Grupo 2: representan a los responsables del contenido intelectual o artístico, la producción física y la difusión o la custodia de las entidades del primer grupo. Las entidades del segundo grupo incluye: persona (un individuo), entidad

corporativa (una organización o grupo de individuos y/o organizaciones), y familia.

3. Grupo 3: representan un conjunto adicional de entidades que se utilizan como materias de las obras. El grupo incluye: concepto (una idea o noción abstracta), objeto (una cosa material), acontecimiento (una acción o suceso) y lugar (una localización).

Un grupo de trabajo de IFLA, junto con otros organismos, desarrollaron modelos conceptuales para continuar con el trabajo del FRBR y determinaron la necesidad de realizar un análisis más detallado de las entidades que constituyen el centro de atención de las autoridades de materia, tesauros, esquemas de clasificación y de las relaciones entre dichas entidades. Los modelos conceptuales desarrollados fueron:

- Requisitos Funcionales de Datos de Autoridad, en inglés *Functional Requirements for Authority Data* (FRAD): modelo entidad relación para vincular los datos que se almacenan en los registros de autoridad de la biblioteca para las necesidades de los usuarios de estos documentos y facilitar el intercambio de esos datos (IFLA, 2014a).
- Requisitos Funcionales para Datos de Autoridad de Materia, en inglés *Functional Requirements for Subject Authority Data* (FRSAD): modelo entidad relación que representar cómo las entidades (grupo 3 de FRBR) pueden estar vinculadas y controladas dentro del universo bibliográfico. El modelo está destinado a facilitar el intercambio global y uso de los datos de autoridad de materia (IFLA, 2014b).

El metamodelo CRADLE

El metamodelo CRADLE (Cooperative-Relational Approach to Digital Library Environments) está basado en un lenguaje visual para la definición de conceptos y servicios relacionados con el desarrollo de bibliotecas digitales, de acuerdo con un metamodelo (Malizia et al., 2010). Una colección de herramientas permite la generación automática de varios servicios, que se definen con el lenguaje visual y las interfaces

gráficas de usuario que proporcionan el acceso. Las entidades básicas del CRADLE son los actores, servicios, colecciones, documentos, y estructuras:

1. Entidad actor: son los usuarios de las bibliotecas digitales. Los actores interactúan con la BD a través de servicios (interfaces) que son (o pueden ser) afectados por las preferencias de los actores y los mensajes (los eventos provocados). En el metamodelo CRADLE un actor es una entidad con un comportamiento que al mismo tiempo puede generar eventos.
2. Entidad de servicios: describe los escenarios, actividades, operaciones y tareas que en última instancia, especifican las funcionalidades de una BD, como la recolección, la creación, difusión, evaluación, organización, servicios de personalización, la preservación, entre otros.
3. Entidad colección: son los grupos de documentos de tipo arbitrario (por ejemplo, los bits, personajes, imágenes, etc) que se utilizan para modelar el contenido estático o dinámico.
4. Entidad documento: son los elementos básicos en la BD y se modelan con la etiqueta de los atributos y la estructura.
5. Entidad de estructura: es un elemento que especifica una parte de un todo. Ejemplo: hipertextos, taxonomías, las relaciones entre los elementos o de contención.

#### *5.1.2.2 Modelos de datos para repositorios institucionales*

##### *DSpace*

DSpace es una herramienta de código abierto desarrollada por el Instituto Tecnológico de Massachusetts (MIT) en colaboración con HP para la implementación del repositorio del MIT (DSpace, 2014). DSpace fue liberada en el 2002 y se presenta como una solución que proporciona toda la funcionalidad necesaria de un repositorio digital, lo cual permite la administración de colecciones digitales tales como libros, artículos, fotos, videos, tesis y entre otros (Tansley, Bass, & Smith, 2003). Los datos son organizados como ítems que pertenecen a una colección que a su vez pertenece a una comunidad. Según OpenDOAR (2014), en su plataforma están registrados 2.600 repositorios y el 41,7% (1085) usa DSpace, siendo el software líder para repositorios.

Esta herramienta está constituida sobre una arquitectura de tres capas, formada por módulos y componentes independientes, lo que favorece un encapsulamiento de la funcionalidad. El núcleo principal está conformado por los módulos ubicados en la ***capa de negocio***, encargado de negociar la gestión de los contenidos de los archivos junto con sus usuarios, autorizaciones y flujos de trabajos (workflows). La inclusión de funciones en la ***capa de aplicación***, se hace gracias a add-ons (programas que solo funcionan anexados a otros). La comunicación entre los distintos componentes de la aplicación se realiza con el mundo exterior a través de la API pública de DSpace. La ***capa de almacenamiento***, usa una base de datos relacional (PostgreSQL o en Oracle) para almacenar todo el contenido de la organización, los metadatos de los contenidos, la información sobre los usuarios de los archivos, las autorizaciones y las diferentes etapas de los workflows que se están ejecutando. Al igual que la capa anterior, se cuenta con una API para esta capa.

Cada capa es invocada por otra capa. Por ejemplo, la capa de aplicación no puede ser usada directamente por la capa de almacenamiento. Aunque la lógica para las acciones de autorización está en la capa de negocio, el sistema se basa en las aplicaciones individuales en la capa de aplicación para una autenticación correcta y segura de los usuarios de archivos. La razón de esta elección de diseño es que los métodos de autenticación pueden variar ampliamente entre las diferentes aplicaciones, así que tiene sentido dejar la lógica y la responsabilidad en esas aplicaciones. El código fuente está organizado para respetar en forma estricta la arquitectura de tres capas y sólo a través de los métodos de la API pública de un componente se les da acceso.

DSpace tiene como entidad básica el ítem, el cual contiene metadatos y contenidos digitales que son definidos como bitstreams que son clasificados como bundles del ítem. La estructura interna de un ítem es expresada por metadatos estructurales, los cuales definen las relaciones entre las partes constitutivas. DSpace está soportado por colecciones y comunidades que pueden tener una o más colecciones, donde un ítem pertenece a una sola colección. Los metadatos adicionales a la estructura predefinida son representados como bitstreams y son almacenados en el sistema de archivos (filesystem). Cada bitstream tiene asociado un formato de bitstream específico con niveles de soporte para asegurar la preservación. Los archivos almacenados

cambian su nombre de acuerdo con la propia estructura interna y son ubicados en la carpeta "assetstore" con el metadato relacionado que debe ser almacenado en la base de datos.

*EPrints*

EPrints (2014) es un sistema de software gratuito para repositorios que en su origen fue utilizado por diversas comunidades universitarias para la creación de repositorios institucionales, es decir, la gestión de publicaciones científicas que no son modificadas frecuentemente. Nació de la idea de Stevan Harnad en el 2000, y se desarrolló en el Departamento de Ciencias de la Computación y Electrónica de la Universidad de Southampton, en el Reino Unido, por Christopher Gutteridge, con la colaboración de Mike Jewell. Según OpenDOAR (2014), EPrints representa el 14,2% de los repositorios registrados.

La entidad básica de EPrints es el objeto digital, el cual es un registro que contiene metadatos, donde uno o más documentos (archivos) pueden ser enlazados con el objeto digital que tendrá con un identificador único. EPrints no considera las colecciones porque agrupa los objetos de datos por campos específicos (materia, año, título, etc). Asimismo, no existe una definición de relaciones entre documentos, solamente usa URLs en campos de metadatos específicos.

En cuanto a los metadatos, son definidos por el administrador. El objeto digital contiene metadatos que son almacenados en una base de datos relacional (MySQL/Oracle) y los documentos del objeto digital se guardan en el sistema de archivos (en el sistema operativo sobre el cual esta operando), los cuales se preservan idénticamente a la forma como se ingresaron a través del software, es decir, sin alterarse.

*Fedora*

Su acrónimo en inglés es *Flexible Extensible Digital Object Repository Architecture* y se le conoce como Fedora Commons; ofrece a los usuarios solamente los servicios de un repositorio, es decir, permite almacenamiento y gestión de los objetos digitales, y a los programadores le ofrece librerías para la gestión del repositorio (Fedora, 2014). Tiene

una arquitectura modular basada en el principio de que la interoperabilidad y extensibilidad para una mejor integración de datos, interfaces, y módulos, aunque no tiene una interfaz *front-end*. Esta característica no lo hace muy adecuado para crear repositorios "listos para usar" como lo son DSpace o EPrints. Fedora posee una arquitectura de gestión de activos digitales (Digital Asset Management, DAM) sobre la cual se pueden construir muchos tipos de bibliotecas digitales y/o repositorios institucionales. Esta plataforma de software comenzó en 1997 como un proyecto de investigación de Carl Lagoze y Sandy Payette en el Grupo de Digital Library Research Group, financiado por DARPA y la NSF en la Universidad de Cornell, donde se construyó la primera implementación de referencia y otra técnica, basada en CORBA. Según OpenDOAR (2014) el 1,6% (33) usa Fedora.

Fedora es un sistema flexible basado en los principios de la arquitectura orientada a servicios con un alto grado de abstracción y una arquitectura basada en modelos de objetos digitales. Proporciona una capa de gestión de propósito general para objetos digitales. La gestión de objetos se basa en modelos de contenido que representan objetos de datos (unidades de contenido) o colecciones de objetos de datos. Los objetos contienen enlaces entre fuentes de información (datastreams), metadatos internos o externos, metadatos de sistema, y comportamientos (procesos de software que pueden ser usados con los datastreams). El sistema es escalable y flexible lo que le permite con proyectos de repositorios externos o distribuidos. La estructura interna es determinada por Fedora Object XML (FOXML), el cual está basado en METS. La arquitectura del servidor Fedora tiene cuatro APIs (Application Programming Interfaces) principales: gestión, acceso, búsqueda y obtención de metadatos. Al igual que DSpace los documentos cambian sus nombres de acuerdo con la estructura interna y son almacenados en la carpeta "data/datastream". Su sistema de gestión de base de datos es relacional, generalmente se usa MySQL, Oracle o PostgreSQL.

*El modelo BIBFRAME*

BIBFRAME (LC, 2014) se enfoca en el uso de la web semántica, los principios y mecanismos de Linked Data y el Resource Description Framework (RDF), para poder desarrollar un modelo básico de datos. La *Library of Congress* de Estados Unidos se ha

propuesto repensar y analizar el esquema bibliográfico vigente, a partir de la migración del formato MARC 21 a este modelo que proponen. Ellos tienen como objetivo la evolución hacia un nuevo esquema que deberá adaptarse mejor a las futuras necesidades para representar la información bibliográfica en la web. Sus trabajos comenzaron en el 2012 pero aún se encuentra en las fases iniciales de consolidación.

El modelo BIBFRAME consiste en 4 clases principales:

- Trabajo creativo *(creative work)*: un recurso que refleja una esencia conceptual del recurso catalogación.
- Instancia (i*nstance*): un recurso que refleja un individuo, es la materialización de la obra.
- Autoridad (*authority*): un recurso que refleja los conceptos de autoridad clave que han definido las relaciones reflejadas en el trabajo y en la instancia. Por ejemplo, recursos de autoridad que incluyen personas, lugares, temas, organizaciones, etc.
- Anotación (*Annotation*): un recurso que mejora nuestro conocimiento sobre otro recurso al conocer, como mínimo, "quién" está haciendo la anotación.

Estos recursos de información pueden ser reensamblados dentro de una arquitectura coherente que permita la catalogación cooperativa a un nivel mucho más detallado que antes. Luego, a medida que se aproveche la web como una arquitectura para los datos, se podrán realizar cambios a estos recursos (por ejemplo, alguien agrega nueva información sobre una persona, nuevas asignaciones relacionadas con la materia, etc), eventos de notificación pueden aparecer para actualizar automáticamente los sistemas que hacen referencia estos recursos. Además, estos archivos de información pueden ser utilizados de manera más efectiva a un nivel granular y proporcionar una información importante en las colecciones locales, colecciones especiales y datos de terceros, y pueden ayudar a contextualizar el contenido de la biblioteca (Kroeger, 2013).

*5.1.2.3. Síntesis*

En líneas generales, existe diversidad de modelos conceptuales y modelos de datos para Bibliotecas Digitales y/o Repositorios Institucionales, los cuales se han desarrollado en

los últimos 15 años aproximadamente. Se centran, en forma general, en un sistema con servicios, roles de administración, usuarios y recursos, sin compatibilidad entre ninguno de ellos. Asimismo, se relevaron una diversidad de estándares que dan soporte al marco de referencia propuesto.

Esta investigación, se enfoca en el **Recurso**, el elemento central de un repositorio. De tal manera que el propósito es obtener una representación de recursos a través de un marco de referencia provisto por los modelos descriptos y no en construir un repositorio institucional. Por ello, la comparación no tiene cabida en esta indagación porque los modelos expuestos estan vinculados a las funcionalidades y no a la representación del recurso porque ninguno se enfoca en la representación. A continuación se sintetizan las características generales de los modelos conceptuales y modelos de datos:

1. Características de los modelos conceptuales
    - Conocidos cómo modelos de referencia.
    - Indican como gestionar un depósito de recursos para garantizar y asegurar el acceso y preservación a esos recursos.
    - Estandarizan los conceptos, terminologías y relaciones del dominio.
    - Definen procesos desde la incorporación hasta la disponibilidad de los recursos.
    - Buscan establecer un diagrama entendible del sistema para que las partes involucradas que intervienen en el repositorio comprendan el contexto.
2. Características de los modelos de datos
    - Basados en modelos conceptual de bases de datos.
    - Tienen arquitecturas de software propias.
    - Diseñan estructuras propias de información y de bases de datos.
    - Organizan la información identificada (modelo conceptual) dentro de un marco lógico.

Es importante mencionar que se conocen otros modelos que tienen menos presencia en los repositorios en el mundo (OpenDOAR, 2014):

- DC-Library Application Profile (DC-Lib) (ASIS&T, 2014a).

- UK Office for Library and Information Networking (UKOLN, 2014).
- Data Model's Digital Asset Management System at University of Texas (DAMS, 2014).
- Variations2 (IU, 2014).
- DC Abstract Model (DCAM) (ASIS&T, 2014b).
- OpenURL Framework (Z39.88) (NISO, 2014).
- ABC (Lagoze & Hunter, 2006).
- CIDOC Conceptual Reference Model (CRM, 2014).
- Europeana Data Model (EU, 2014).

### 5.1.3. La Norma ISO 14721

La descripción de los diferentes modelos conceptuales y modelos de datos existentes para el dominio LIS, sirvió para tomar como ejemplo a la Norma ISO 14721 en la implementación del marco de referencia propuesto, en esta investigación. Tal decisión tiene su base en seguir un estándar reconocido en este dominio que permita mantener conceptos básicos para hacer una implementación y además toma en consideración los 4 elementos de una representación de recursos (ver sección 5.1.1.).

Esta norma conocida como el modelo de referencia OAIS (Open Archival Information System), con su primera edición en el 2003, y en junio de 2012 una segunda edición (CCSDS, 2012). Este estándar modela las partes que componen un sistema abierto de archivo de información resumido en la Figura 5-2 (por ejemplo un repositorio institucional), sin definir cómo debe ser su implementación ya que la misma dependerá del usuario, como es el caso de esta investigación. Las funciones principales del modelo OAIS son conservar la información, garantizar el acceso y asegurar la preservación. La norma recomienda un entorno representado por:

- Un productor: es quien provee la información para ser preservada y gestionada, pueden ser personas o sistemas de software.
- Un consumidor: es quien busca y adquiere información de interés que se ha preservado.
- Un gestor: es el rol desempeñado por aquellos que establecen el conjunto de

políticas del OAIS. No incluye las operaciones diarias.

El modelo OAIS propone un conjunto de seis módulos funcionales conocidos como entidades, que abarcan los diferentes servicios que pueden ser ofrecidos (De Giusti et al., 2012). Para cada uno de estos módulos se definen unos requisitos funcionales así como la forma en que interactúan. El elemento central de la norma es el *Information Package* (IP) que lo conforman el objeto digital y todos sus metadatos, y es la unidad de intercambio del OAIS, pero también es lo que se va actualizando y cambiando según la función que debe cumplir. El IP puede clasificarse en tres subtipos según su función en el proceso de archivo:

- *Archival Information Package* (AIP): contiene, como mínimo, suficiente información de un objeto como para garantizar la preservación a largo plazo. Busca mantener la mayor calidad posible de información descriptiva de preservación y de los objetos representados o contenidos.
- *Submission Information Package* (SIP): es el paquete que proviene del productor y se va a incorporar al OAIS. Suele contener menos información que el AIP.
- *Dissemination Information Package* (DIP): es el paquete que se entrega a un consumidor en respuesta a una solicitud. La información de empaquetado toma muchas formas dado que los usos de OAIS son diversos. Puede ser tan completo como los AIP a partir de los cuales se construye o ser sólo una breve descripción del paquete.


*Figura 5-2. Modelo Funcional OAIS. Fuente: De Giusti et al. (2012)*

El modelo OAIS propone un conjunto de seis módulos funcionales (también conocidos como entidades) que abarcan los diferentes servicios que pueden ser ofrecidos. Para cada uno de estos módulos se definen unos requisitos funcionales así como la forma en que interactúan entre ellos (para mayor descripción de los módulos funcionales ver sección 2.2.3.). En la Figura 5-2 se observan los tres actores (productor, consumidor y gestor) y las seis entidades (carga, almacenamiento del archivo, gestión de datos y acceso, planeamiento de la preservación y administración), y se muestra que el proceso que recomienda la norma puede iniciarse cuando el productor (un actor) suministra el recurso (paquete de entrada) llamado SIP a través de la entidad *ingest*, que luego se convierte en AIP terminando en la entidad de *archival storage*. El flujo puede continuar cuando el consumidor busca una información en el sistema, que es entregada como un DIP a través de la entidad *access*, ya que ha sido preservada en el sistema previamente. Simultáneamente, los datos relacionados con los artículos y el repositorio mismo, se mantienen organizados a través de la entidad de *data management* (gestión de datos). Después, hay una entidad especializada para la administración adjunta a la gestión, que serían los administradores y el responsable del repositorio, relacionándose con las secciones de *gestión de datos* y *planificación de la conservación*. Esto permite una gestión estructural y también ayuda a mantener los AIP a lo largo del tiempo. El módulo de *planificación de la conservación* desarrolla estrategias y normas de conservación, monitorea las últimas novedades y avances en el campo, además de monitorear los cambios en la comunidad designada, para que toda la información nueva que se solicite, se pueda adjuntar a los AIP correspondientes.

En la Figura 5-3, se observa la recomendación que hace la norma sobre lo que debe contener un paquete de información (Information Package – IP) dentro del sistema de archivo de información sin la implementación, que en el contexto de este estudio es tomando como el modelo conceptual a seguir. Todo paquete de información, es llamado en esta tesis **Recurso**, el cual debe tener un contenido de información y una información para la preservación. El contenido de información debe componerse del objeto digital o físico y de la representación de la información. En síntesis, en este estudio cada representación final del IP (recomendado por la norma) consiste en los metadatos de los recursos dentro del modelo en el DSL gráfico desarrollado en fase 1.

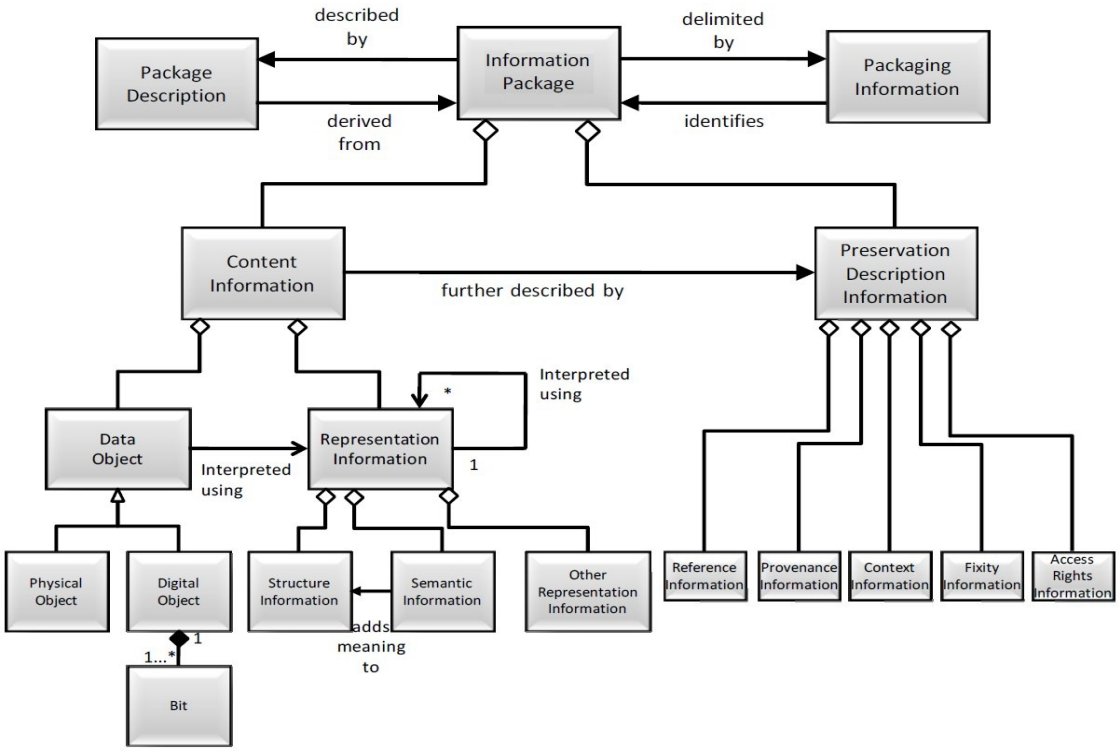


*Figura 5-3. Information Package recomendado por la norma ISO 14721. Fuente: Norma ISO 14721.*

## *5.2. Diseño*

En esta sección se describe y explica el diseño del marco de referencia sobre la base del análisis contextual, relevamiento teórico de los modelos conceptuales y modelos de datos, y la norma ISO 14721. Por tal razón, el diseño partió del modelo conceptual vinculado con modelo IP recomendado por la norma ISO 14721 y termina con la implementación de todo un marco de referencia para el desarrollo de funcionalidades.

### 5.2.1. DSL norma ISO 14721

Un DSL (Domain Specific Language) permite proporcionar al usuario un lenguaje adaptado a la semántica de un dominio y resolver problemas propios de ese dominio, por ejemplo, SQL. Los DSL pueden ser textuales o gráficos. Algunos entornos conocidos para el desarrollo de DSL textuales son: EMFText, Xtext, TCS, etc. y otros entornos para DSL gráficos: GMF y Graphiti. En ambos casos, se definen metamodelos donde se indican las notaciones (textual o gráfica) de cada elemento del lenguaje.

El objetivo de este módulo consistió en proponer un DSL gráfico que permitiera

diseñar un modelo flexible (PIM) que represente los recursos dentro de SEDICI y sirva de modelo para otros sistemas de repositorios, sobre la base de la recomendación de conceptos, relaciones y semántica de la norma ISO 14721 gracias al modelo IP observado en la Figura 5-3. Este proceso también es conocido como DSM (Domain Specific Modeling), porque permite el modelado entre las partes involucradas en la administración del repositorio bajo la misma (terminología).

Para la construcción de un editor de modelos (el DSL) se debe tener un metamodelo que soporte el diseño (ajustado al metametamodelo Ecore) de acuerdo con las especificaciones del EMP. El DSL permite obtener un modelo basado en la sintaxis permitida por el metamodelo (representado por un archivo XMI) a partir de una sintaxis abstracta y al menos una sintaxis concreta.

| **CU-01** | **Fase 1 - DSL** |
|---|---|
| Versión | 1.0 (01/05/2014) |
| Precondición | Conocer y entender la norma ISO 14721 |
| Descripción | Implementar un lenguaje gráfico propio del dominio, que permita a las partes interesadas desarrollar un modelo independiente de la plataforma para representar los recursos del repositorio institucional de la UNLP, SEDICI |
| Secuencia normal (pasos) | 1. Definir el metamodelo que ajuste el modelo flexible de representación de recursos de SEDICI.<br>2. Generar la estructura de clases que dan soporte al modelo del dominio.<br>3. Definir los elementos gráficos a mostrar y seleccionar el marco de trabajo del editor, es decir, la sintaxis concreta.<br>4. Relacionar los elementos anteriores y construir el editor de modelos para el metamodelo seleccionado.<br>5. Modelar en una nueva sesión de Eclipse. |
| Postcondición | Integrar el modelo generado en el proyecto de la fase 2 |
| Excepciones | Ninguna |
| Comentarios | Las partes interesadas: dueños del negocio, expertos del dominio y desarrolladores, encuentran en este DSL un punto de acuerdo, para generar un modelo flexible que sirva de base para construir una aplicación para la representación de recursos en SEDICI. |

*Tabla 5-2. Caso de uso de la Fase 1 – DSL*

La sintaxis abstracta define los diferentes elementos del lenguaje y las reglas que establecen cómo pueden ser combinados y la sintaxis concreta (una o más) define cómo

los elementos del lenguaje aparecen en una notación textual o gráfica. Cada sintaxis concreta se necesita establecer una relación (mapping) con la sintaxis abstracta. Un tipo de sintaxis concreta es la sintaxis de serialización que tiene como objetivo lograr la persistencia e intercambio de las especificaciones expresadas con el lenguaje, no está destinada para ser usada por las personas, es decir, una sintaxis concreta está destinadas para ser usada por personas y otras no. Otro ejemplo de sintaxis concreta es la desarrollada en este módulo. En otras palabras, la sintaxis abstracta del DSL es el metamodelo y la sintaxis concreta es la notación textual o gráfica para lograr el modelo.

En la Tabla 5-2 se presenta el caso de uso de la fase 1, donde se describen las actividades a realizar. Brooks (1987) señala que la parte más difícil al implementar un sistema informático es precisamente conocer qué se debe construir. Por tal razón, la exposición de los casos de uso ayudará a las partes involucradas a implementar las fases deseadas.

### 5.2.2. Del modelo flexible al modelo relacional

La fase anterior (el DSL) genera como salida un modelo (en formato XMI) ajustado a las reglas y relaciones definidas en el metamodelo `SimpleClass.ecore`, que según la OMG (2014) representa el PIM del marco de referencia propuesto. Por ello, se requiere trasladar el modelo obtenido del DSL a un modelo relacional (PSM) que luego permita representar en forma persistente los recursos de SEDICI diseñados en la fase anterior, el cual debe estar ajustado a un metamodelo relacional, llamado `SimpleRDBMS.ecore`. En la Figura 5-4 se aprecia el esquema de la transformación M2M (modelo a modelo) realizada, es decir, pasar un PIM (modelo flexible) a un PSM (modelo relacional). Tanto el PIM como el PSM deben encontrarse bajo los lineamientos (léxico, sintaxis y semántica) de un lenguaje acorde con un metalenguaje, que para el enfoque Model-Driven sería el metamodelo y el metametamodelo. En cuanto a la transformación, ese código (o lenguaje) debe entender las reglas de los lenguajes A y B donde el lenguaje de transformación debe estar acorde a un metalenguaje. Dentro del contexto del Model-Driven, este lenguaje de transformación recibe como entrada los metamodelos que entienden los lenguajes A y B, haciendo transparente para el usuario el proceso de transformación, ya que son los ingenieros de

software los encargados de diseñar dichos lenguajes (metamodelos) y metalenguajes (metametamodelos).


*Figura 5-4. Esquema de transformación M2M*

| CU-02 | Fase 2 - M2M |
|---|---|
| Versión | 1.0 (01/05/2014) |
| Precondición | Modelo generado en la fase 1, ajustado al metamodelo `SimpleClass.ecore`. Asimismo, se debe contar con un metamodelo destino, que en este caso es `SimpleRDBMS.ecore`. Conocimiento sobre ATL, el lenguaje de transformación. |
| Descripción | Para una transformación de modelo a modelo, es necesario que ambos estén ajustados a un metamodelo, es decir, a una sintaxis abstracta, para que el lenguaje intermediario pueda entender tanto la entrada como la salida. |
| Secuencia normal (pasos) | 1. Modelo de entrada, `norma.XMI`<br>2. Metamodelo de salida, `SimpleRDBMS.ecore`<br>3. Reglas, condiciones y helpers en ATL<br>4. Ejecutar la transformación del modelo de entrada al modelo destino. |
| Postcondición | Copiar modelo generado y pegarlo en el proyecto de la fase 3 (independientes) |
| Excepciones | De existir entidades aisladas, no serán transformadas a entidades destino. |
| Comentarios | ATL sirve de base para las transformaciones M2M dentro del contexto de EMP. Ambos metamodelos usados en toda la investigación, parten de ejemplos y tutoriales que pone a disposición Eclipse y ATL. |

*Tabla 5-3. Caso de uso de la Fase 2 – M2M*

Para lograr la transformación de la Figura 5-4 se seleccionó el paquete de software ATL, que consiste en un lenguaje que soporta procesos M2M y comprende metametalenguajes como Ecore. A continuación el caso de uso de la fase 2 (Tabla 5-3).

### 5.2.3. Modelo relacional a script de creación

El objetivo de este módulo consiste en realizar una transformación del modelo relacional a texto (M2T) para crear la base de datos de la propuesta. Este script sirvió de base para establecer los requerimientos de incorporación de recursos a la base de datos y que serán expuestos en la siguiente fase 4. Para el modelo de entrada (relacional), obtenido en la fase de transformación M2M, se seleccionó el paquete de software Acceleo (lenguaje que soporta procesos M2T) que permite interpretar un modelo de entrada (ajustado a un metamodelo) para obtener un texto deseado. En la Figura 5-5 se aprecia en forma abstracta el proceso de transformación M2T deseado. En la Figura. el PSM esta escrito sobre el metamodelo SimpleRDBMS, capaz de comprender el modelo dicho PSM. Este metamodelo está acorde al metametamodelo llamado Ecore. La transformación está destinada a traducir el modelo PSM en código (un script SQL), es decir, dicha transformación (M2T) la realiza Acceleo comprendiendo las reglas del modelo PSM para llevar a un código de texto (SQL) según las reglas definidas por el ingeniero de software. Para implementar este módulo fue necesario crear el caso de uso (Tabla 5-4):

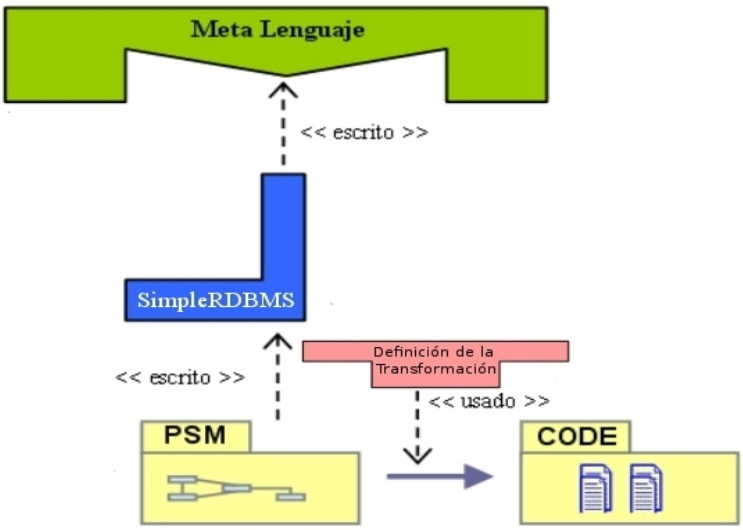


*Figura 5-5. Esquema de transformación M2T*

| CU-03 | Fase 3 - M2T |
|---|---|
| Versión | 1.0 (01/05/2014) |
| Precondición | Modelo relacional de la fase 2 (ajustado a `SimpleRDBMS.ecore`), necesario para generar la transformación de acuerdo con la gramática de la plataforma específica seleccionada. Conocimiento sobre Acceleo, lenguaje para transformaciones M2T. |
| Descripción | Esta transformación, permite pasar un modelo específico a un documento en formato de texto, acorde con la gramática de ese lenguaje seleccionado. Por eso, el modelo relacional `basededatos.XMI` se convirtió en el archivo de texto `crear.SQL` |
| Secuencia normal (pasos) | 1. Seleccionar el metamodelo para el modelo de entrada `basededatos.XMI`.<br>2. Generar un editor a partir del metamodelo.<br>3. Abrir una sesión de Eclipse desde el editor generado.<br>4. Definir el código que entenderá el modelo de entrada y lo convertirá en el archivo de texto para crear una base de datos relacional.<br>5. Ejecutar la transformación M2T.<br>6. Crear la base de datos en PostgreSQL. |
| Postcondición | El documento en formato de texto de salida, `crear.SQL`, debe ubicarse en el proyecto de la fase 4. |
| Excepciones | Ninguna |
| Comentarios | La transformación M2T se realiza en otra sesión de Eclipse, la cual se crea para que el editor entienda la sintaxis abstracta definida en el metamodelo y el proyecto Acceleo que se crea pueda identificar el metamodelo de entrada. |

*Tabla 5-4. Caso de uso de la Fase 3 – M2T*

### 5.2.4. Incorporación de registros

El script de creación de la base de datos sirvió de base para establecer los lineamientos de incorporación de registros en la propuesta, ya que definió cuáles erán las tablas y campos permitidos para ingresar la información que proviene del DSpace-SEDICI.

Este proceso se pudo diseñar bajo el enfoque Model-Driven, pero debido al nivel de detalle y múltiples entradas de archivos, se desarrolló una rutina en Java para leer la información de entrada (varios archivos de textos) y generar un script de salida en formato de texto, que representó las diferentes tuplas o registros de recursos que provienen de SEDICI para el sistema propuesto en la investigación. El caso de uso por

el que se rigió para la fase 4 es expuesto en la Tabla 5-5.

| CU-04 | Fase 4 - Incorporación |
|---|---|
| Versión | 1.0 (01/05/2014) |
| Precondición | Script de creación de la base de datos obtenido en la fase anterior y los siguientes archivos con la información de SEDICI:<br>1. Recursos exportados por DSpace, `METS.xml`<br>2. Comunidades<br>3. Colecciones<br>4. Autores<br>5. Instituciones<br>6. Revistas<br>7. Números de revistas |
| Descripción | Esta fase se implementó a través de un proyecto Java que comprende los diferentes archivos de entrada de información y los ajustes a las diferentes tablas, campos y relaciones que provienen de `crear.SQL` de la fase anterior. Luego, esta fase entrega un archivo de salida llamado `inserts.SQL` |
| Secuencia normal (pasos) | 1. Leer archivo `crear.SQL`<br>2. Leer archivos de SEDICI: recursos, comunidades, colecciones, autores, instituciones, revistas y números de revistas.<br>3. Ejecutar la rutina de transformación en Java para obtener el archivo `insert.SQL`<br>4. Correr el script de incorporación en la base de datos creada en la fase anterior. |
| Postcondición | Ninguna |
| Excepciones | Ninguna |
| Comentarios | En esta fase, no se aplicó ningún enfoque Model-Driven por la complejidad de ajustar múltiples archivos de entrada a un metamodelo. |

*Tabla 5-5. Caso de uso de la Fase 4 – Incorporación*

### 5.2.5. Aplicaciones

Después de obtener un modelo propio que representó los recursos de SEDICI y se llevó a una base de datos para hacerlo persistente en el tiempo, se generó una aplicación web para visualizar los recursos obtenidos y exportar los recursos del RI, es decir, tener un marco de referencia que permita la generación de una aplicación web para implementar servicios que normalmente no se encuentran o difieren de lo deseado en diferentes

sistemas de repositorios. Por ello, en esta fase 5, las aplicaciones web son una de las vías para poder evaluar todo el planteamiento como una solución al problema expuesto en esta investigación, a través de la implementación de funcionalidades y/o aplicaciones propias para el dominio de los RI gracias al marco de referencia propuesto.

De manera tal que se buscó una herramienta bajo los mismos lineamientos expresados a lo largo del estudio que permitió la generación de aplicaciones bajo el enfoque Model-Driven. Esa herramienta se llama WebRatio, en ella se puede sincronizar una base de datos creada, desarrollar un prototipo navegacional de la aplicación deseada, y después, generar la aplicación sin intervenir en la generación de código. El caso de uso para la fase 5 es el presentado en la Tabla 5-6.

| **CU-05** | **Fase 5 - Aplicaciones** |
|---|---|
| Versión | 1.0 (01/05/2014) |
| Precondición | Base de datos creada en PostgreSQL con el contenido importado desde SEDICI, tal y como se comentó en la fase anterior. |
| Descripción | WebRatio, permite construir aplicaciones web a partir de una base de datos ya creada junto con un modelo navegacional, por ello, en esta fase se sincronizó la base de datos creada en los módulos 3 y 4, luego, se generan las aplicaciones web deseadas tomando como base el modelo navegacional diseñado. |
| Secuencia normal (pasos) | 1. Sincronizar base de datos en PostgreSQL<br>2. Crear el modelo navegacional para mostrar los recursos importados.<br>3. Crear el modelo navegacional que permita exportar los recursos deseados de acuerdo con las recomendaciones de la norma ISO 14721 y criterios configurables.<br>4. Construir las aplicaciones web a partir de los modelos navegacionales. |
| Postcondición | Ninguna |
| Excepciones | Ninguna |
| Comentarios | WebRatio resume el enfoque Model-Driven a través de la metodología WebML |

*Tabla 5-6. Caso de uso de la Fase 5 – Aplicaciones*

## *5.3. Implementación*

Una vez diseñado el planteamiento de la propuesta estructurada en 5 módulos o fases en la sección anterior, el siguiente paso fue implementar dicha estructura para evaluar los resultados y sus beneficios. El metametamodelo (capa M3) usado fue Ecore que provee

la plataforma EMP y se encuentra en el paquete `org.eclipse.emf.ecore` de Eclipse. Ecore es la especificación más alta que existe en la arquitectura de capa MDA (M3) y sobre ella se ajustan los metamodelos y se construyen los modelos del proyecto (Vogel, 2014).

La versión de la plataforma de software Eclipse Modeling Tools que se utilizó fue la 2.0.2. que se encuentra dentro del entorno de desarrollo usado, Eclipse Kepler 4.3. Para la utilización de Ecore en Eclipse fue necesario tener instalado el plugin de EMF. Este plugin provee básicamente dos herramientas para construir un modelo basado en Ecore, una el *Ecore Model* que es un editor manual y funciona en un estilo de árbol de navegación; la otra, es el *Ecore Diagram* siendo éste un editor gráfico, similar a las herramientas gráficas, para la creación de diagramas de clases UML. Cualquiera de las dos formas se usa para crear el diagrama basado en Ecore (en formato XMI). En esta indagación se reutilizó el metamodelo que provee Eclipse a través de sus diferentes tutoriales y ejemplos, llamado `SimpleClass.ecore`, el cual se puede observar en la Figura 5-6 y se agregan los tipos de datos Entidad Abstracta (EA) y vocabularios. Asimismo, para las transformaciones M2M se usó ATL 3.4.0., para transformaciones M2T se usó Acceleo 3.1.2., para la construcción del DSL se utilizó GMF (Graphical Modeling Framework) 3.1.0. y para las diferentes rutinas Java se usó Java SE Runtime Enviroment 1.7. Todos estos lenguajes, herramientas y librerías se usaron bajo el sistema operativo Ubuntu 12.04.4 LTS.

De igual manera, se empleó el entorno de desarrollo de software Eclipse para el desarrollo de la aplicación web con el objetivo de visualizar los recursos y exportar dichos recursos, a través del proyecto WebRatio, que permite implementar una aplicación web bajo el enfoque Model-Driven. La versión usada fue WebRatio 7.2, la cual se usó a través de la licencia académica de WebRatio s.r.l.

Todo el código fuente de los diferentes módulos se encuentra en un proyecto GitHub que lleva por título Tesis (GitHub, 2014b). El hecho de encontrarse dentro de una estructura GitHub lo hace disponible y reutilizable por la comunidad, bajo los mismos términos legales, es decir, se encuentra bajo una Licencia Creative Commons Atribución-NoComercial-CompartirIgual 4.0 Internacional.

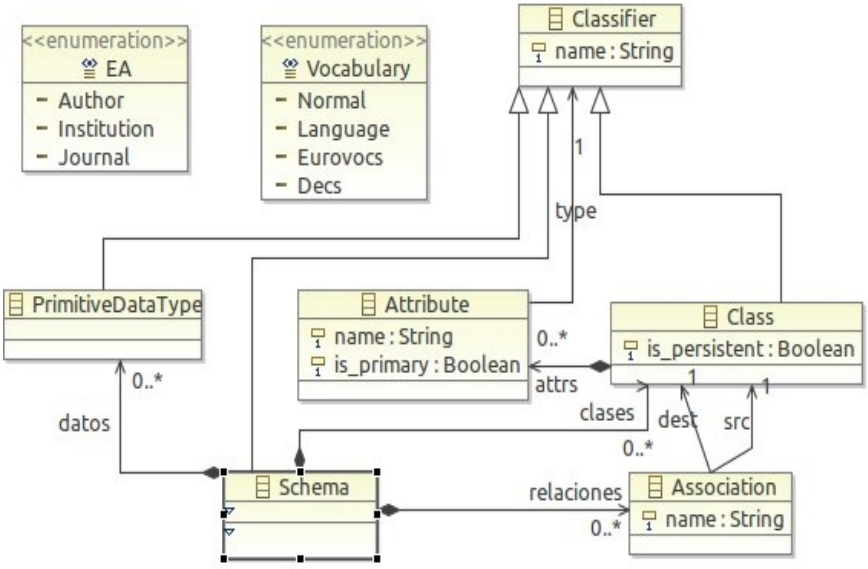


*Figura 5-6. Metamodelo SimpleClass.ecore*

La implementación consistió en desarrollar el DSL que permitió a las partes interesadas construir un modelo de representación de recursos, que luego de una serie de transformaciones se obtuvo una base de datos, que permitió desarrollar algunas funcionalidades especificas para el dominio de los RI.

### 5.3.1. DSL norma ISO 14721

La construcción del editor se centró en el metamodelo SimpleClass que se encuentra en los diferentes tutoriales del EMP[2]. A partir de este metamodelo, que corresponde a la sintaxis abstracta, se definieron los conceptos, relaciones y la semántica deseada. El funcionamiento del DSL es sencillo, para crear entidades se deben arrastrar los nodos de la barra de herramientas al frame de trabajo, completar los campos y relacionarlos respetando las siguientes reglas:

- Una entidad puede ser un contenido de información, una descripción de la información (metadatos) o una entidad abstracta.
- Sólo se permiten tipos de datos definidos como datos primitivos.
- Las relaciones sólo pueden corresponder entre las entidades.
- Cuando se transforme el modelo a un modelo relacional, las entidades que estén sin alguna relación no serán tomadas en cuenta.

[2] http://www.eclipse.org/atl/atlTransformations/SimpleClass2SimpleRDBMS/README.txt

Este editor se construyó bajo el componente de software GMF Tooling (Graphical Modeling Framework Tooling) de EMP que hace parte del proyecto GMP (Graphical Modeling Project) (Eclipse, 2014b). Por ello, a través del GMF se construyó la sintaxis concreta gráfica. El proceso para la construcción del DSL gráfico se visualiza en la Figura 5-7 y es una guía para el desarrollo de herramientas de este tipo (Steinberg et al., 2008). Todo el código fuente correspondiente a este módulo se encuentra en el proyecto GitHub Tesis/Fase1 (GitHub, 2014b).

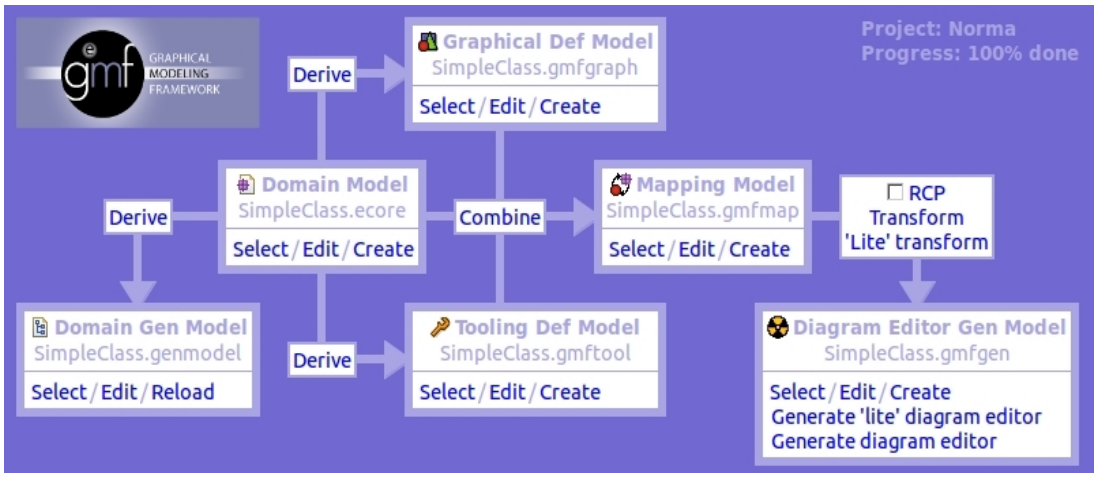


*Figura 5-7. GMF (Dashboard)*

Según el Dashboard de la Figura 5-7, lo primero que se debe crear es el *Domain Model*, que corresponde al metamodelo `SimpleClass.ecore`, es decir, este metamodelo es la sintaxis abstracta del DSL. El siguiente paso es realizar el *Domain Gen Model*, que es un modelo que permite transformar automáticamente el modelo Ecore a un código fuente Java. El código se generó aplicando patrones de transformación. El resultado es un conjunto de clases Java, que son utilizados más adelante en la herramienta (Ver Figura 5-8).

Posteriormente, se creó el *Graphical Def Model*, usado para definir los elementos gráficos como figuras, nodos y conexiones, los cuales se van a usar cuando se diseñe el modelo (Figura 5-9), es decir, `Norma.XMI` que es el modelo obtenido del DSL. El resultado es un archivo con los elementos nodo, etiqueta del diagrama y conexión, para representar los temas del *Domain Model*, tales como canvas, etiquetas y conexiones.

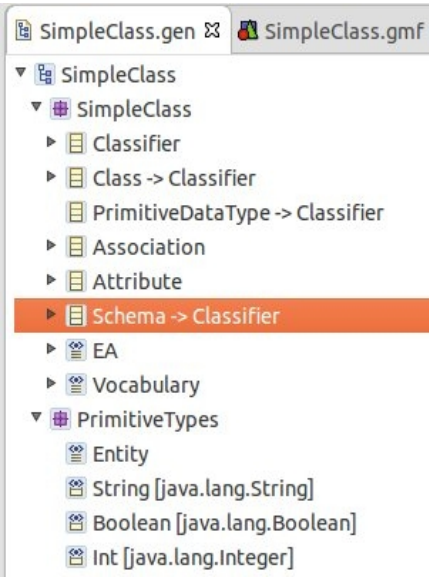


*Figura 5-8. Domain Gen Model*


*Figura 5-9. Graphical Def Model*

El siguiente paso de acuerdo con el Dashboard (Figura 5-7) es la creación del *Tooling Def Model*, este es usado para especificar la paleta (Palette) o barra de herramientas de creación, acciones, etcétera, para los elementos gráficos (Figura 5-10). Existe un elemento en el nivel superior "Tool Registry" en el que se encuentra una paleta. La "Palette" contiene un "Tools Group" con elementos de tipo "Creation Tool" correspondiente a los nodos tema y conexiones para elementos de subtemas.


*Figura 5-10. Tooling Def Model*

El *Mapping Model* (Figura 5-11) es el siguiente paso en el Dashboard que combina los tres modelos: el *Domain Model* (la sintaxis abstracta), el *Graphical Def Model* (la sintaxis concreta) y el *Tooling Def Model* (el modelo de herramientas).

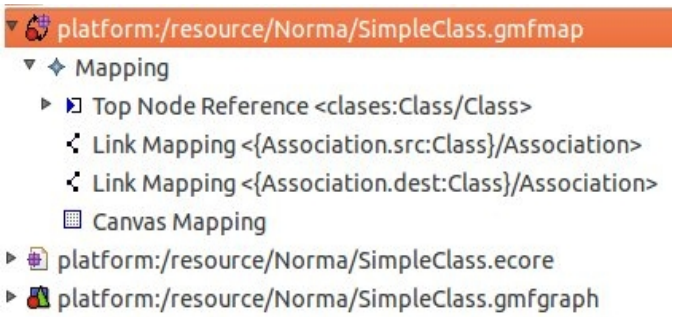


*Figura 5-11. Mapping Model*

El último paso es la creación del *Diagram Editor Gen Model*, aquí se establecen las propiedades para la generación de código para el editor, ver Figura 5-12, similar al *Domain Gen Model*. A partir de este modelo se obtiene un plugin para Eclipse que contiene la herramienta DSM, es decir, es donde se presenta el área de trabajo (frame o canvas) que permite elaborar el modelo (`Norma.XMI`). La apariencia de la herramienta se muestra en la Figura 5-13.

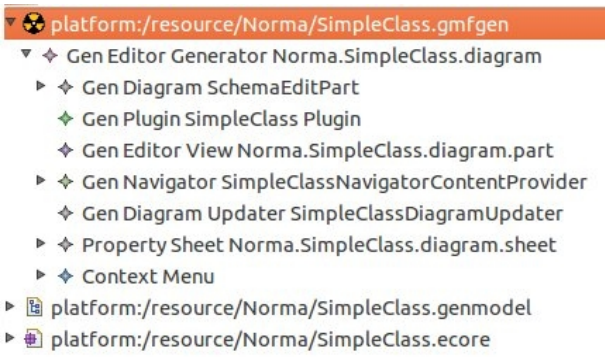


*Figura 5-12. Diagram Editor Gen Model*

En resumen, en esta sección se describió la creación de una herramienta DSL gráfica (M1), basada en un metamodelo (M2) y que a su vez se ajusta al metametamodelo Ecore (M3). El siguiente paso es convertir este modelo, que se obtuvo con la herramienta DSL, en un modelo relacional a través de una transformación M2M.

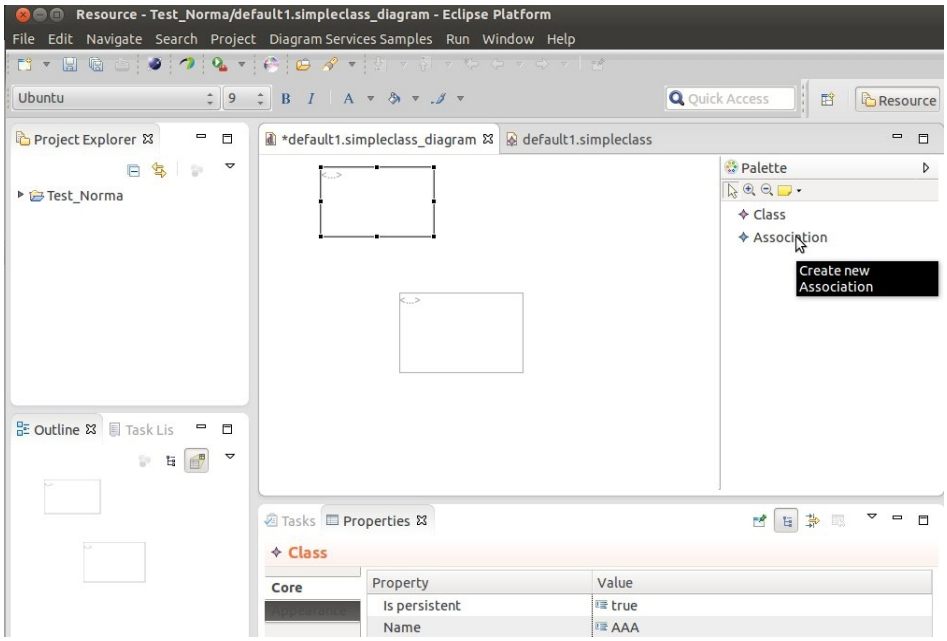


*Figura 5-13. Editor del modelo* `norma.XMI`

### 5.3.2. Del modelo flexible al modelo relacional

Para la transformación M2M se usó ATL, que provee una interfaz fácil de usar y dentro de la plataforma EMP. Se requiere definir cuáles son los metamodelos para los modelos de entrada y salida, y luego, definir la transformación según la sintaxis de ATL. La Figura 5-14 muestra el esquema general ATL aplicado.

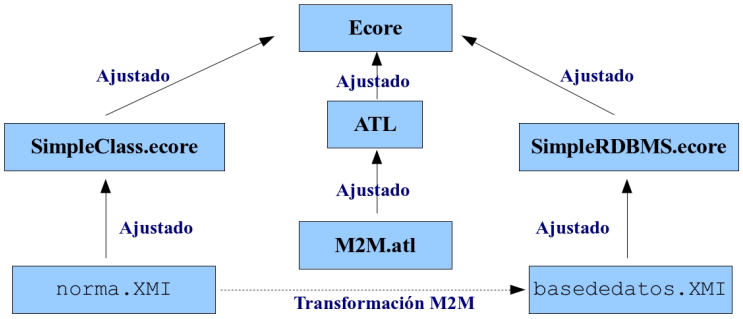


*Figura 5-14. Esquema de transformación M2M*

El código fuente del proyecto ATL, que corresponde a esta fase, se encuentra en el proyecto GitHub Tesis/Fase2 (GitHub, 2014b). Los dos metamodelos que permitieron

la implementación de los modelos de entrada (`norma.XMI`) y salida (`basededatos.XMI`) son tomados como base de los diferentes tutoriales y ejemplos que ofrece la plataforma EMP, es decir, `SimpleClass.ecore` y `SimpleRDBMS.ecore` respectivamente[3]. A continuación se muestra la configuración para ejecutar el proceso ATL (Figura 5-15), en el que se pueden apreciar los metamodelos y modelos de entrada y salida.

Este proceso de ejecución es posible gracias al código ATL, puesto que parte de ese código se observa en la Figura 5-16. Los metamodelos tienen que estar definidos como entrada y salida en el encabezado del código, luego se definen las condiciones, las reglas y los helpers (equivalente a los métodos en Java).

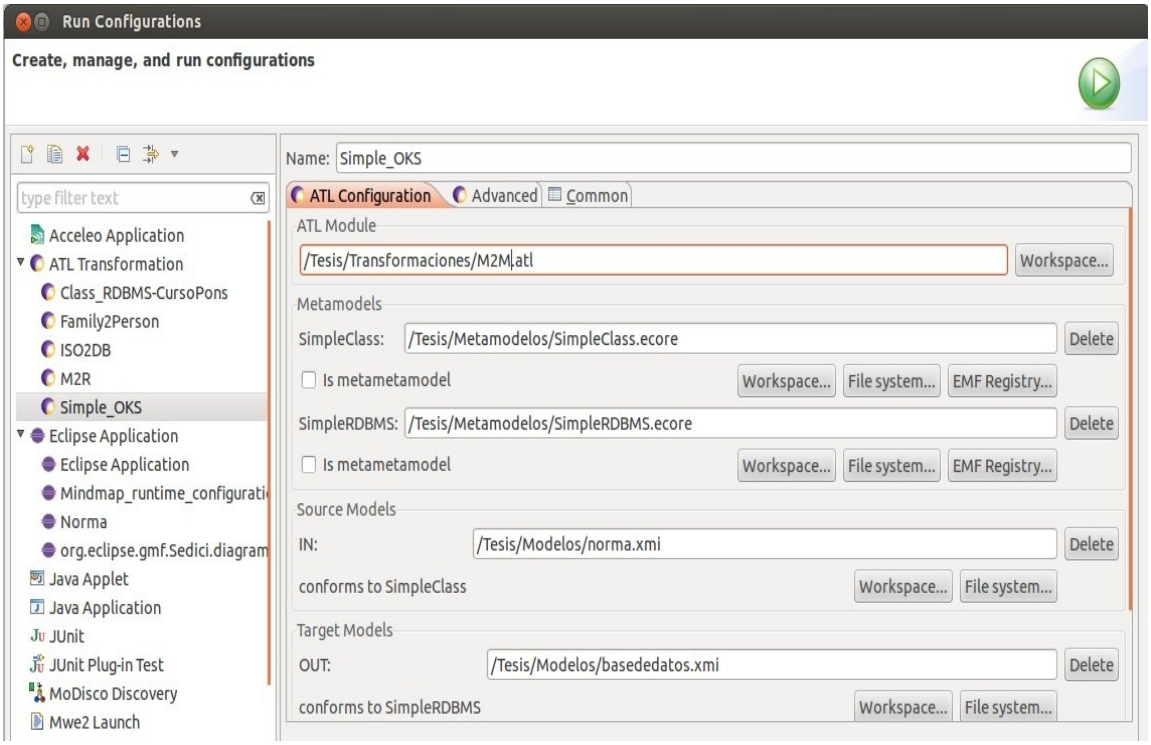


*Figura 5-15. Proceso de ejecución M2M – ATL*

La salida resultante de este proceso es un modelo relacional ajustado al metamodelo SimpleRDBMS, que en este caso se llama `basededatos.XMI`.

[3] http://www.eclipse.org/atl/atlTransformations/SimpleClass2SimpleRDBMS/README.txt

```
ATL_Curso.atl
 2 create OUT : SimpleRDBMS from IN : SimpleClass;
 3
 4 rule PersistentClass2Table{
 5     from
 6         c : SimpleClass!Class (
 7               c.is_persistent and c.parent->oclIsUndefined()
 8             )
 9
10     using {
11         primary_attributes : Sequence(TupleType(name : String,
12                                                 type : SimpleClass!Classifier,
13                                                 isPrimary : Boolean)
14                                ) =
15                     c.flattenedFeatures->select(f | f.isPrimary);
16
17
18         persistent_features : Sequence(TupleType(
19                                               name : String,
20                                               class : SimpleClass!Class,
21                                               offcet : Integer,
22                                               nofAttrs : Integer
23                                             )
24                                    ) =
25             c.flattenedFeatures->iterate(tuple; acc : Sequence(TupleType(name : String,
26                                                                  class : SimpleClass!Class,
27                                                                  offcet : Integer,
28                                                                  nofAttrs : Integer))=Sequence{}
29                         if tuple.type->oclIsKindOf(SimpleClass!Class)
30                         then
```

*Figura 5-16. Código de transformación M2M – ATL*

### 5.3.3. Modelo relacional a script de creación

En esta fase se realizó una transformación M2T con el paquete de software Acceleo y dentro del contexto de la plataforma EMP. Se definió cuál es el metamodelo que se ajusta el modelo de entrada de esta fase, además del código Acceleo (archivo con extensión .mtl) que permitió la transformación del modelo a texto. En la Figura 5-17 se observa el esquema general de la transformación M2T con Acceleo.

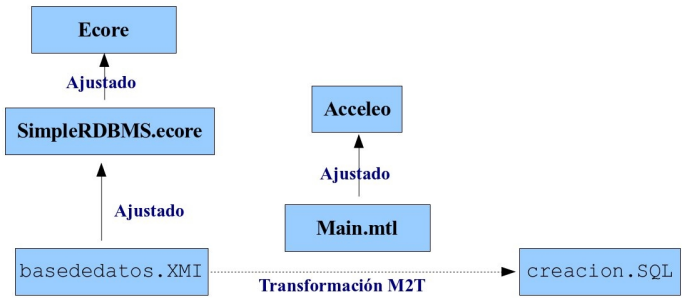


*Figura 5-17. Esquema de transformación M2T*

El código fuente de este módulo se encuentra en el proyecto GitHub Tesis/Fase3 (GitHub, 2014b). Es importante destacar, que primero se creó un proyecto Java, en el que se implementó un modelo generador del dominio para construir un editor en otra sesión de Eclipse, en la que se creó un proyecto Acceleo que reconoce el URI del

metamodelo de entrada que se encuentra en el proyecto Java. Luego, a partir de ese proyecto se inició el proceso de ejecución de la transformación M2T, como se visualiza en la Figura 5-18.

Este proceso transforma el modelo de entrada, `basededatos.XMI`, en un archivo de texto llamado `creacion.SQL`, a partir de código Acceleo que se encuentra en el archivo `generate.mtl`. En la Figura 5-19 se observa el código Acceleo en un frame y el texto de salida obtenido por la transformación en otro frame.

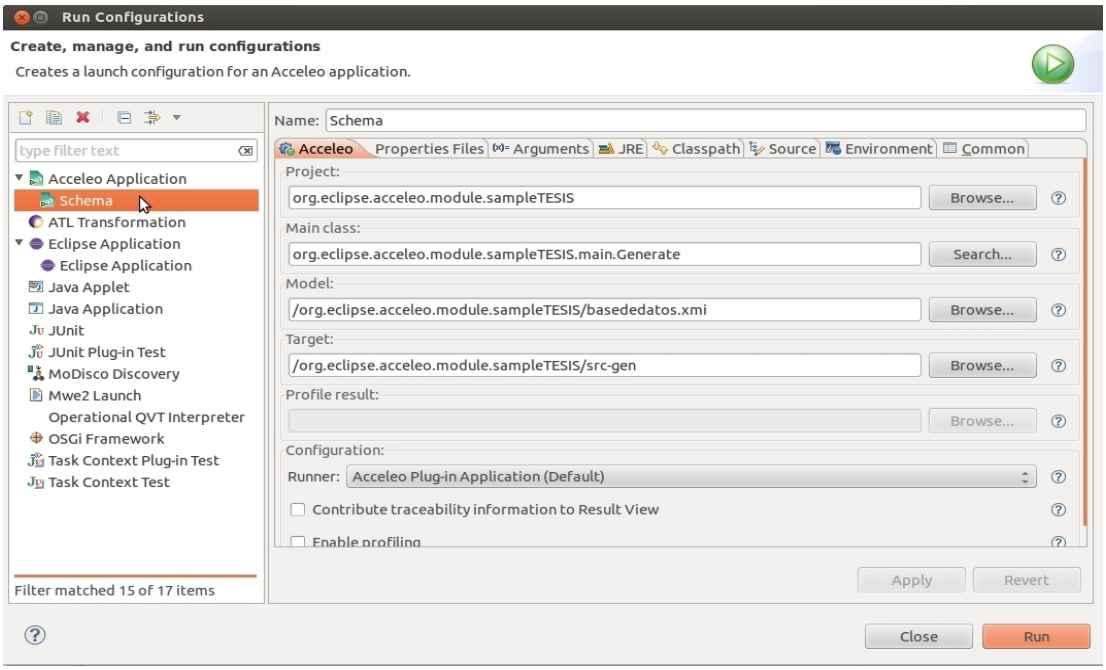


*Figura 5-18. Proceso de ejecución M2T – Acceleo*

### 5.3.4. Script de ingreso de registros

Uno de los objetivos de esta propuesta era representar los recursos de SEDICI a través de un modelo flexible que permitiera desarrollar otros trabajos. Por ello, en esta investigación se acopló la información de los recursos de DSpace-SEDICI al modelo propuesto. En la fase anterior se mostró cómo se creó la base de datos, ahora bien, en esta fase se muestra cómo se incorporó la información que proviene de DSpace-SEDICI:

1. Recursos de SEDICI en paquetes de información conocidos como AIP (según norma ISO 14721), en archivos XML bajo el esquema de metadatos METS.

2. Comunidades desde una base de datos en MySQL.
3. Colecciones desde una base de datos en MySQL.
4. Autores desde una base de datos en MySQL.
5. Instituciones desde una base de datos en MySQL.
6. Revistas de las revistas desde una base de datos en MySQL.
7. Números de las revistas desde una base de datos en MySQL.

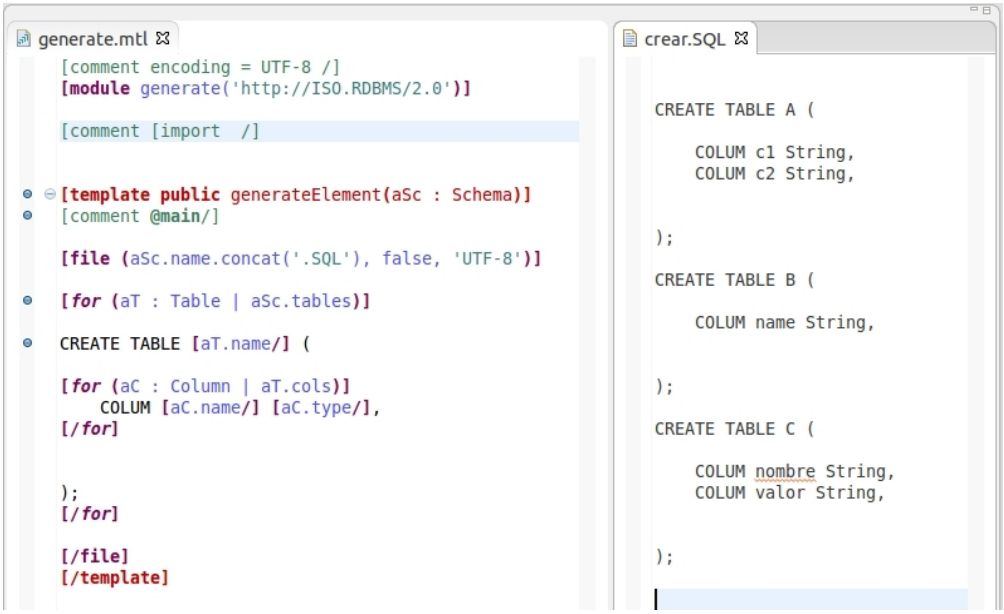


*Figura 5-19. Código de transformación M2T – Acceleo*

Se desarrolló un proyecto en Java que transformó toda la información (varios archivos) provenientes de SEDICI al modelo relacional de acuerdo con los lineamientos recibidos del script de creación de la base de datos obtenida en la fase anterior. El resultado de este módulo es un script SQL de incorporación de información a la base de datos llamado `insert.SQL`. En la Figura 5-20 se observa parte del código Java que permitió esta transformación llamada `T2T_insert.java`.

El código fuente correspondiente a este fase se encuentra en el proyecto GitHub Tesis/Fase4 (GitHub, 2014b).

T2T_insert.java

```
package t2t;
import javax.xml.parsers.DocumentBuilderFactory;

public class T2T_insert {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        try {
            File fXmlFile = new File("schema_db.xmi");
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
            Document doc = dBuilder.parse(fXmlFile);
            doc.getDocumentElement().normalize();
            System.out.println("Root element :" + doc.getDocumentElement().getNodeName());
            NodeList nList = doc.getElementsByTagName("SimpleRDBMS:Table");
            System.out.println(nList.getLength());
            for (int temp = 0; temp < nList.getLength(); temp++) {
                Node nNode = nList.item(temp);
                System.out.println("\nCurrent Element :" + nNode.getNodeName());
                if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                    Element eElement = (Element) nNode;
                    for (int j = 0; j<eElement.getElementsByTagName("cols").getLength(); j++)
                        {
                        System.out.println("\t"+
                                eElement.getElementsByTagName("cols").item(j).getAttributes().getNamedItem("name").getNodeValue
                                );
```

*Figura 5-20. Código de transformación T2T – inserts*

### 5.3.5. Aplicaciones

La evaluación del planteamiento realizado se hizo mediante la implementación de una aplicación que visualizó todos los recursos obtenidos y el modelo. Para ello, se seleccionó a WebRatio bajo el enfoque Model-Driven que permitió construir una aplicación web a partir de una base de datos existente, la se sincronizó con la plataforma de WebRatio. A continuación, se elaboró un modelo navegacional que sirve de base para generar la aplicación web deseada. El código fuente de todo este módulo se encuentra en el proyecto GitHub Tesis/Fase5 (GitHub, 2014b).

Las aplicaciones web desarrolladas fueron dos: una, para visualizar todos los recursos importados desde DSpace-SEDICI adaptados al modelo propuesto, y, la otra aplicación, que permite exportar los recursos deseados en formato XML y ajustados a criterios similares (pero configurables) de las recomendaciones de la norma ISO 14721, es decir, exportar un paquete de información similar al AIP. A continuación se observan dos Figuras (5-21 & 5-22) que representan parte del modelo navegacional y parte de la aplicación web generada a partir de dichos modelos.

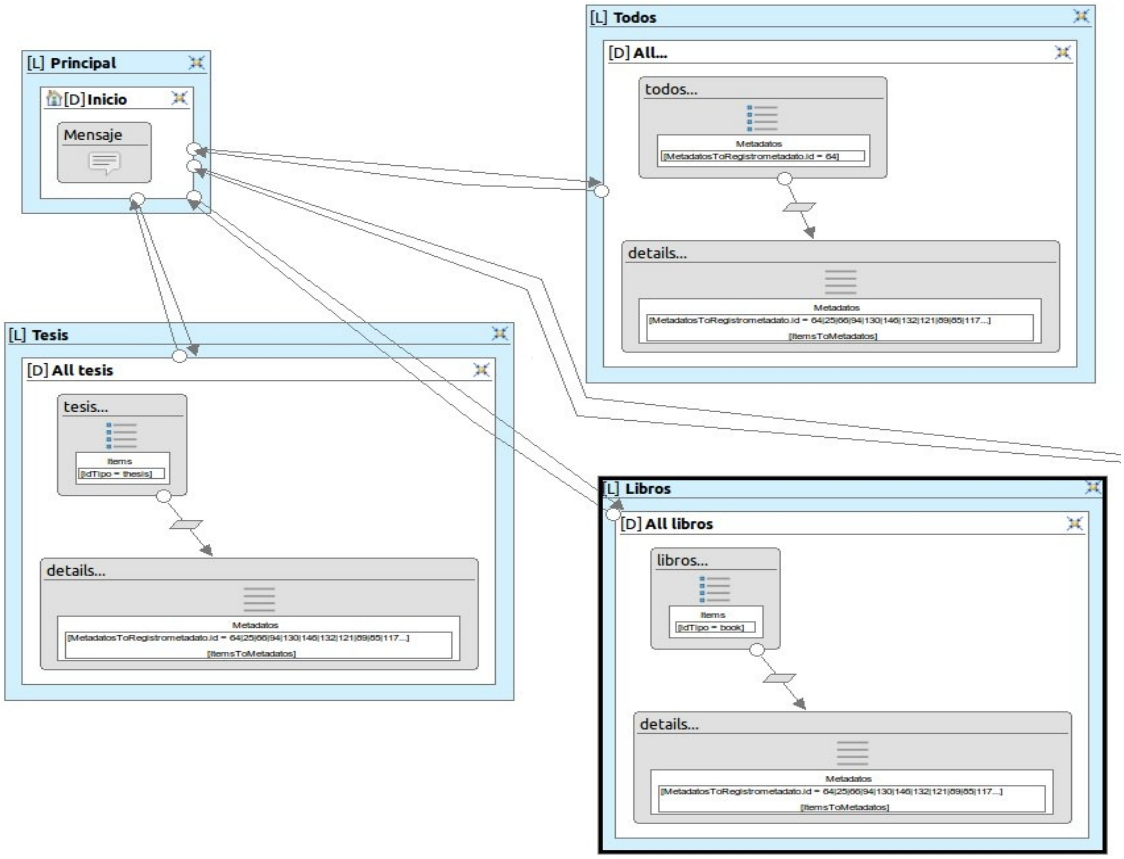


*Figura 5-21. Modelo navegacional – WebRatio*

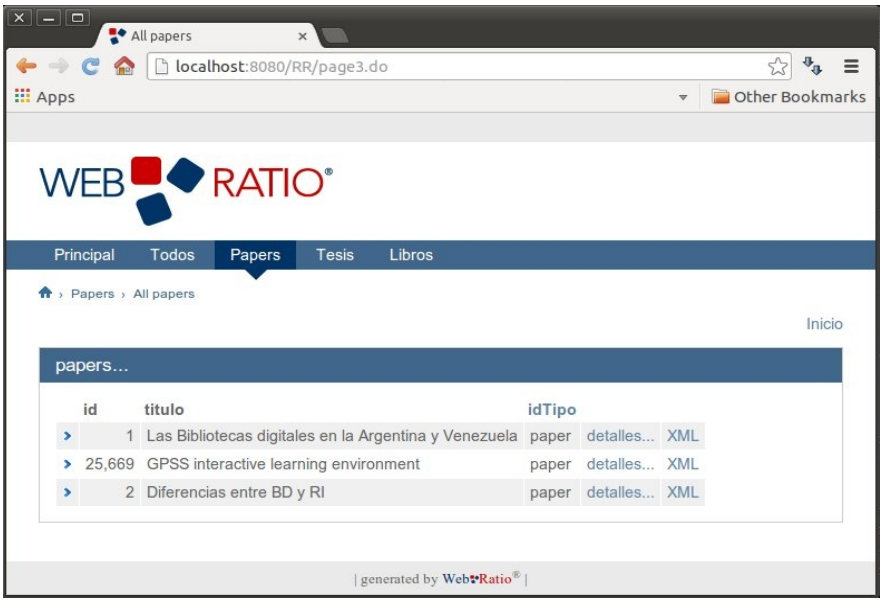


*Figura 5-22. Capa de presentación – WebRatio*

El modelo navegacional y la aplicación no son posibles sin un modelo relacional, es decir, una base de datos que soporte y mantenga persistentemente la

información importada. La Figura 5-23 muestra la base de datos resultante a partir del modelo generado por el DSL en la Fase 1, el cual ha evolucionado a través de transformaciones M2M, M2T y T2T, hasta llegar a una base de datos en PostgreSQL para ser reconocida y sincronizada en WebRatio, y a partir de ella desarrollar las aplicaciones web antes descriptas.

Para exportar los recursos como archivos XML que representan un tipo de AIP configurable, fue necesario definir un XML Schema Definition (XSD) en concordancia con la base de datos, ya que el mismo permitió hacer el mapeo para extraer la información y convertirla en un AIP, definido con reglas propias. A continuación se expone el XSD, `sedici.xsd` (Tabla 5-7):

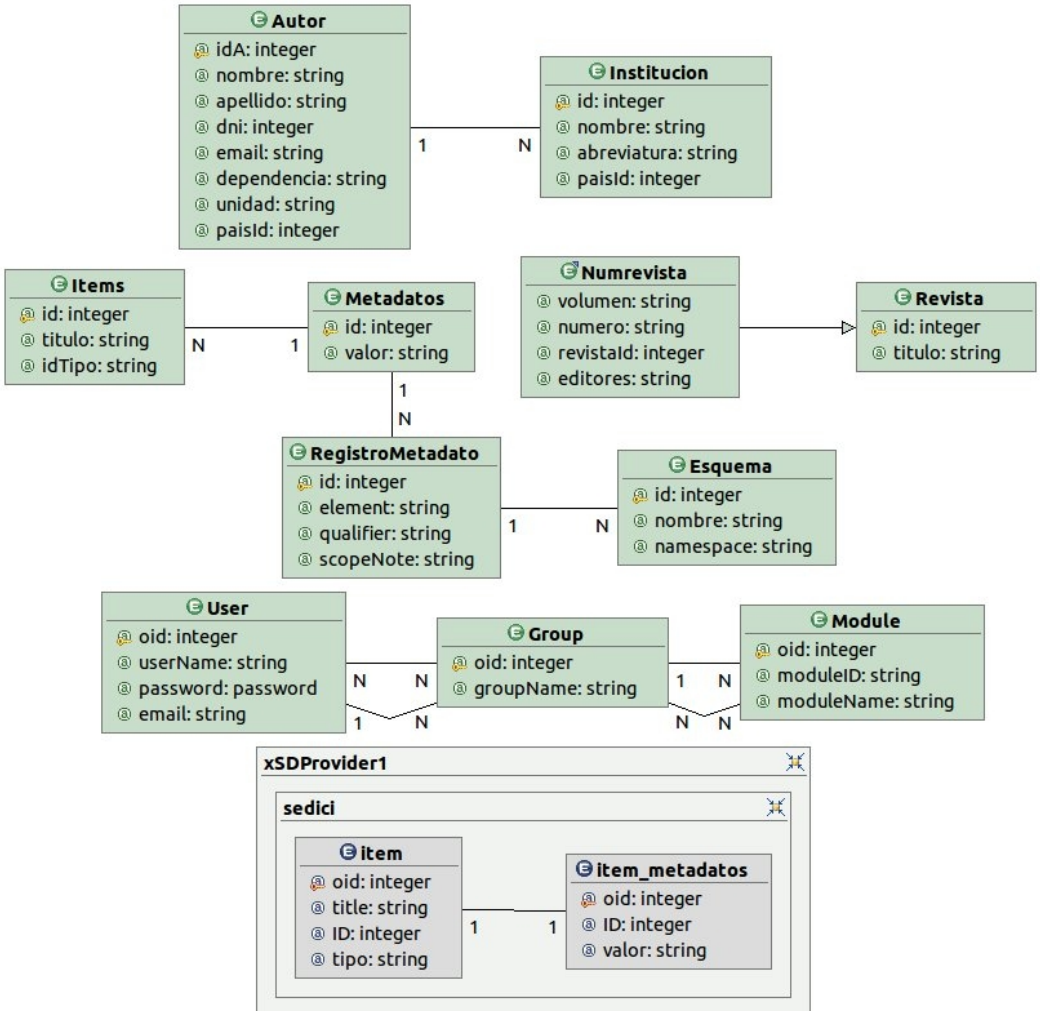


*Figura 5-23. Modelo relacional – WebRatio*

```
<?xml version="1.0" encoding="ASCII"?>
<xsd:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
version="1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="item">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="title" type="xsd:string" />
        <xsd:element name="id" type="xsd:int" />
        <xsd:element name="tipo" type="xsd:string" />
        <xsd:element name="metadatos">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="id" type="xsd:int" />
              <xsd:element name="valor" type="xsd:string" />
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

*Tabla 5-7. XSD en WebRatio*

## *5.4. Resultados*

### 5.4.1. DSL norma ISO 14721

El construir un DSL para el dominio de los repositorios, de acuerdo con las recomendaciones de la norma ISO 14721, permitió a las partes interesadas hablar un mismo idioma para el diseño de un sistema de gestión de repositorios institucionales a partir de una representación de los recursos. Por ello, en la fase 1 las partes interesadas desarrollaron un modelo similar al empaquetado de información (IP) recomendado en la norma ISO 14721 que se apreció en la Figura 5-2 y como resultado se obtuvo una base de datos similar a la expuesta en la Figura 5-23. De igual forma, en la Figura 5-24 se observa el modelo resultante del DSL.

Por tanto, gracias al modelo obtenido a través del DSL, los *dueños del negocio* entiendieron el sistema propuesto manteniendo la funcionalidad y coherencia, ya que se construyó una propuesta a partir de la recomendación de la norma, los *expertos del negocio* se satisficieron porque pudieron participar en el diseño de una propuesta basada en los mismos conceptos y relaciones. Además lograron usar la norma en sus diferentes tareas, y los *desarrolladores de software* se concentraron en las condiciones impuestas en el modelo generado con el DSL, ya que este modelo les provee mayor concordancia con los requerimientos del sistema.

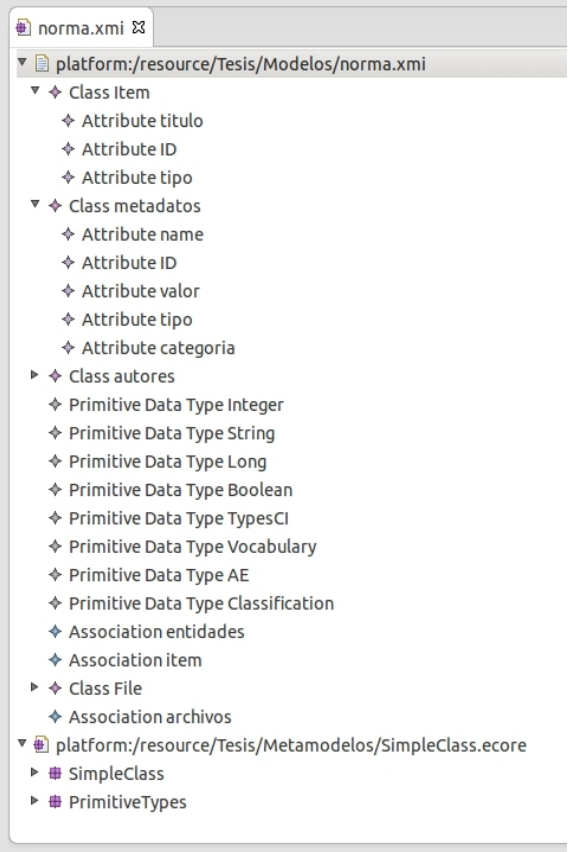


*Figura 5-24. Modelo obtenido del DSL*

### 5.4.2. Del modelo flexible al modelo relacional

Mediante el desarrollo de un DSL que permitió generar un modelo flexible de acuerdo con los lineamientos de la norma ISO 14721, es decir, un modelo independiente a la plataforma (PIM) represento el punto de partida de toda la propuesta. Este PIM se transformó en un modelo específico de la plataforma (PSM) que fue un modelo relacional para una base de datos (en PostgreSQL). Este modelo es una instancia del metamodelo específico `SimpleRDBMS.ecore`.

Al ejecutar la transformación M2M se logró trasladar los conceptos y relaciones que pertenecen al dominio de los repositorios institucionales a una plataforma específica para ser implementada y utilizada por los desarrolladores de software y por los usuarios. Estos contarán con los servicios tradicionales y servicios nuevos generados a partir del modelo relacional hecho realidad con la base de datos. Este proceso de transformación es llevado a cabo a partir de las reglas definidas a través del lenguaje ATL.

### 5.4.3. Modelo relacional a script de creación

La fase anterior dispuso un modelo relacional específico (PSM) que se concretó en un documento en formato texto necesario para crear la base de datos PostgreSQL. En otras palabras, no se necesitó un metamodelo destino porque el resultado fue solo texto, es decir, una sintaxis que entiende PostgreSQL, base de datos en la que se encuentra DSpace-SEDICI.

Esta transformación M2T generó el texto de forma explícita; si bien hubiese podido realizarse como una transformación M2M hubiese resultado menos intuitiva y más complicada. Además, esta solución requeriría el diseño de un metamodelo para dicha salida. El beneficio directo es la rápida implementación de una solución que se adapta a cualquier gramática deseada como salida, es decir, se podría requerir una base de datos MySQL, DB2 u Oracle, en vez de PostgreSQL, que a pesar de ser similares sus sintaxis poseen elementos particulares. También, si se desea contar con una base de datos en otro paradigma (ya sea orientada a objetos o documental), se puede realizar alterando el metamodelo de destino en la fase anterior y luego en esta fase de transformación M2T, se define la salida de acuerdo con el lenguaje de transformación Acceleo y la gramática del paradigma deseado.

### 5.4.4. Script de ingreso de registros

La diversidad de información que provienen de múltiples archivos de DSpace-SEDICI para generar un solo documento en formato texto, que permita incorporar esta información de los recursos de SEDICI a la base de datos creada en la fase anterior, hizo que se diseñara un código Java para leer las entradas y poder unificar en una sola salida.

Por ello, no se aplicó ningún criterio de Model-Driven en el desarrollo de esta fase por no ser una solución intuitiva ni fácil de diseñar. El trabajo se limitó a la programación tradicional estructurada de múltiples entradas de archivos y una salida, con un código intermedio que interpretara ambas partes.

### 5.4.5. Aplicaciones

La metodología desarrollada en WebRatio se puede aplicar sobre cualquier repositorio, ya que es posible sincronizar y modificar el esquema de la base de datos del repositorio

y realizar el diseño del modelo de hipertexto que después genera la aplicación web. Este módulo se centró en implementar dos funcionalidades a partir del modelo relacional: una, la visualización de los recursos y la otra, la exportación de dichos recursos. El modelo relacional obtenido podría integrarse en forma bidireccional con otros repositorios.

Otra de las ventajas es que se cuenta con un entorno de desarrollo integrado sobre la base de Eclipse y soportada por la metodología WebML; los paradigmas de construcción de software se ven beneficiados al contar con herramientas de software amenas y en sintonía con las nuevas tecnologías. De igual manera, WebRatio cuenta con una página web que orienta su uso, ofrece un foro para consultas, una wiki con los principales problemas categorizados por nivel de dificultad y un canal de YouTube, en caso de querer profundizar en soluciones más detalladas.

## *5.5. Implicación*

El entender la representación de recursos dentro de un RI y sus diferentes modelos conceptuales y de datos permitió relevar uno de tanto problemas presentes en este dominio la representación de recursos. Se desarrolló un marco de referencia para abordar el problema en cinco fases que estuvieron enmarcados bajo el proceso básico de desarrollo de software. Cada una de las fases resultaron beneficiosas para los tres actores presentes del dominio de los repositorios (desarrolladores, dueños del negocio y expertos del dominio) gracias a aplicación del paradigma Model-Driven así como también de su implementación en el contexto de los repositorios. Se mencionaran cuales fueron los beneficios para los actores:

En cuanto a los *los desarrolladores y/o diseñadores:* responsables de plasmar en código ejecutable los requerimientos del sistema a desarrollar.

- Se obtuvo un menor número de líneas de código escritas, ya que los niveles de abstracción de Model-Driven a través de los modelos y metamodelos diseñados fomento el reuso del código y de los modelos. Además minimizó la tasa de errores y simplificó las tareas las tareas de mantenimiento y mejoramiento.
- Se logro un alto nivel de abstracción para escribir aplicaciones y artefactos de

software a través de la arquitectura de niveles del metamodelado y las capas de modelado de MDA. Este beneficio favoreció el diseño de una aplicación o artefactos de software partiendo de lo más general a lo más concreto, es decir, de forma independiente de la tecnología. Por ejemplo, los objetos del repositorio (documentos, autores, instituciones, tesauros, sistemas de clasificación, áreas temáticas, etc.) se pueden abstraer a un formato general propio de los RI, además de establecer relaciones entre ellos.

- Se posibilitó una especificación de requisitos de usuario a varios niveles mediante un sistema flexible a los cambios. La flexibilidad de estos sistemas se observa si la funcionalidad que se desea agregar es posible implementarla a través de los modelos correspondientes y, evidentemente, en el código generado.
- Se evitó la adopción de una única tecnología de hardware particular gracias a los niveles de abstracción presentes, por tanto, los metamodelos se conviertieron en el eje central, porque representaron el modelo del sistema de manera independiente a la tecnología.
- Se propició la interoperabilidad entre los objetos en los sistemas de RI en un entorno multiplataforma. Esto se logró por los niveles de abstracción que permitieron representar en un modelo general las relaciones entre objetos del RI en diversas plataformas.

Para los *dueños del negocio:* encargados de coordinar y/o financiar el proyecto de desarrollo e implementación del sistema dentro de la organización o institución, los beneficios son:

- El desarrollo de componentes de software para los sistemas de repositorios. Esto se hizo evidente por las fases del ciclo de vida MDD, desde el CIM a la generación de código, ayudando a los desarrolladores a crear diseños de artefactos correctos sin modificar el sistema general (propenso a errores), con el fin de incrementar la productividad y calidad del mismo. Esos artefactos pueden servir como indicadores para evaluar el uso del sistema, por ejemplo, recursos descargados y visitados, tiempo de permanencia en vistas de los objetos, etc.
- La preservación digital de los recursos y/o de los objetos generando estrategias que parten de los niveles altos de abstracción, como son los PIM. La importancia

de la preservación radica en garantizar la perpetuidad en el tiempo del recurso almacenado.

- La generación de código para plataformas previamente especificadas a través de la funcionalidad que ofrece el paradigma, ya que los artefactos de software se generan en los PSM que luego se transformarán en el código deseado.
- La reducción de costes en el desarrollo de aplicaciones debido a la disminución del recurso humano requerido, de las horas hombre y del tiempo invertido en las diferentes actividades relacionadas.
- La documentación de todo el proceso de desarrollo de software, representado básicamente en el modelo PIM, documentación de alto nivel que se necesita para cualquier sistema de software.

Respecto a los *expertos del dominio:* representan a los especialistas presentes en las fases del mundo de los RI los beneficios son:

- Permitir la revisión de modelos por parte de los distintos expertos del dominio. A diferencia de los desarrolladores quienes, se concentran en los detalles técnicos.
- Generación de lenguajes específicos del dominio en las fases de la implementación de los RI bajo Model-Driven, tales como: modelo de datos, modelo de la arquitectura, modelo de las entidades abstractas, interfaz de usuario, entre otros. Este beneficio se obtuvo en el desarrollo de modelos correspondientes a diferentes puntos de vista para las fases principales del paradigma, PIM y PSM, al mismo tiempo la posibilidad de implementar lenguajes específicos del dominio para esos puntos de vista garantizando el reuso de los conceptos.
- Interoperabilidad entre los distintos modelos PSM, principalmente, ya que pueden pertenecer a distintas tecnologías (plataformas). Esta interoperabilidad se logra a través de puentes construidos por las herramientas de transformación de modelos garantizando los conceptos definidos.

# CAPÍTULO 6
# CONSIDERACIONES FINALES

## *6.1. Conclusiones*

Esta investigación se planteó como objetivo general "*proponer un marco de referencia que permitiera definir un modelo general y flexible para obtener la representación de recursos del Repositorio Institucional de forma independiente de las tecnologías usadas*", de manera tal que para dar respuesta a tal propósito se vincularon premisas devenidas de tres disciplinas: Ciencias de la Información, Ciencias Documentales y Ciencias de la Computación (LIS). Entretejer tales áreas en una propuesta relacionada con repositorios institucionales representó un aporte en un área de vacancia en la literatura. En las siguientes líneas se presentan las conclusiones en relación con cada uno de los objetivos específicos planteados inicialmente en este estudio:

- El relevamiento de la literatura sobre las bibliotecas digitales y los repositorios institucionales permitió identificar las funciones básicas para visualizar el problema de la representación de recursos. Para ello, se implementó un marco de referencia que posibilitó el desarrollo de funcionalidades en el dominio de los Repositorios Institucionales (RI) a partir de la representación de recursos, tomando como caso de estudio al Servicio de Difusión de la Creación Intelectual de la Universidad Nacional de La Plata. Los productos obtenidos dentro del marco de referencia fueron: un DSL gráfico para obtener una representación de recursos, 3 transformaciones (M2M, M2T y T2T) a partir del modelo obtenido en el DSL hasta llegar a la base de datos de la representación de datos deseada y una propuesta de generación de funcionalidades dentro del contexto planteado.
- Las diferentes propuestas de repositorios analizadas a partir de sus modelos de datos (DSpace, Eprints, Fedora, etc.) y modelos conceptuales (Norma ISO 14721, 5S, DELOS, FRBR, etc.) sirvieron como guía teórica para la representación de los sistemas de repositorios como un todo. Ninguno de los

modelos reseñados presentaban a los recursos como elemento central, de manera tal que para esta investigación centrada en obtener una representación de recursos y no en diseñar un repositorio institucional, dichos modelos fueron un insumo para comprender las propuestas existentes.

- Los factores que incidieron en los problemas de la representación de recursos en los RI se compilaron en siete ejes de análisis: recursos, esquemas de metadatos, almacenamiento, catalogación, incorporación de recursos, gestión de los datos y acceso. Identificar tales ejes junto con la descripción del problema de la representación de recursos, incidió en la caracterización de la tipología de recursos, esquemas de metadatos y entidades abstractas, y los procesos funcionales de acuerdo con la norma ISO 14721. Esta caracterización junto con la identificación de las funciones básicas de los RI permitieron describir las causas del problema para proponer una solución a la situación planteada en esta investigación. Las principales causa destacadas fueron: diversidad de plataformas de software, diferentes modelos conceptuales y modelos de datos en los cuales sirven de base para repositorios institucional y no se centran en la representación de recursos, varias tipologías de recursos, esquemas de metadatos, recomendaciones de almacenamientos de recursos, la preservación de recursos, reglas de catalogación, entre otras.
- El desarrollar un modelo general que representó los recursos de un repositorio institucional bajo un enfoque Model-Driven mostró cómo las diferentes reglas de catalogación pueden convivir en esta propuesta de investigación en el tiempo, puesto que las reglas se deben representar en el DSL y tienen como objetivo identificar una característica-propiedad para asignarle un valor, luego la semántica de cada regla está dada por las condiciones sobre las que se deben asignar dichos valores y los niveles de granularidad permitidos.
- Se emplearon diversos lenguajes con distintos propósitos (Java, Acceleo, ATL, OCL, Ecore/MOF, WebRatio, SQL, XMI, XML) que hicieron de la propuesta una alternativa compleja y disímil bajo el enfoque Model-Driven. A pesar de eso, se obtuvo un marco de referencia para diseñar un modelo para la representación de recursos. Se entiende que la propuesta se diseñó desde el

punto de vista de los programadores, sin embargo, los dueños del negocio y los expertos del dominio pudieron participar en la primera fase de la propuesta (DSL), es decir, en el diseño del modelo general e independiente de la plataforma basado en las recomendaciones de la norma ISO 14721. Por tanto, la propuesta contó con dos puntos de vista: uno, desde los programadores que tienen una herramienta para la generación de aplicaciones en el dominio LIS garantizando como punto de partida la preservación de los recursos, y, el otro, desde la perspectiva del usuario final, que pueda hacer uso de dichos recursos a través de aplicaciones que nacen de este marco de referencia.

- La validación de la propuesta se hizo con las personas involucradas, mediante la implementación de dos funcionalidades (en prototipo) para usuarios finales bajo el enfoque Model-Driven, usando la herramienta WebRatio. Para ello, se desarrolló una aplicación para observar y recorrer los diferentes tipos de recursos importados desde SEDICI, y a su vez se diseñó un módulo de exportación de recursos bajo un paquete similar al AIP de la norma ISO 14721, con reglas propias y configurables. En el proceso, se determinó que la metodología de WebRatio pueda ser aplicable sobre cualquier repositorio, porque sincroniza y modifica el esquema de la base de datos del repositorio, que junto con el diseño del modelo de hipertexto se genera la aplicación web.
- Se logró el diseño propio de un modelo general, independiente de la plataforma y flexible para obtener una representación de recursos de SEDICI bajo el enfoque Model-Driven. En otras palabras, un marco de referencia en el que se pueden modificar los modelos iniciales y tener resultados casi inmediatos sin necesidad de hacer grandes cambios en el código generador. Este estudio se centró en SEDICI, que posee la particularidad de gestionar numerosas entidades abstractas y de las múltiples entradas de archivos (comunidades, colecciones, autores, revistas, instituciones). Por tanto, el caso de representación de recursos que se logró es replicable en cualquier otra plataforma de software de repositorios debido al marco de referencia propuesto y a la garantía de la visualización de los recursos deseados en el tiempo de forma independiente.
- El enfoque Model-Driven fue transversal a toda la propuesta presentada sobre la

base de la representación de recursos en los RI/BD y permitió que los expertos del dominio y los dueños del negocio del dominio LIS se concentren en implementaciones de software más formales –consolidación de tales sistemas– cuyos principales beneficiarios serán los usuarios finales a través de los diferentes servicios que son y serán ofrecidos por estos sistemas. Por ello, los resultados obtenidos con esta investigación evidencian la limitada literatura al respecto y demuestran cómo de la interrelación del Model-Driven y del mundo LIS se contribuyó con un área de vacancia que, ciertamente, requerirá de posteriores estudios.

## *6.2. Trabajos futuros*

Sobre la base de los resultados obtenidos surgen algunas líneas de trabajo futuros, tales como:

- El marco de referencia desarrollado en esta investigación podría replicarse otras plataformas de software de repositorios como Eprints o FEDORA, incluso sobre otros repositorios en DSpace, de esta manera se puede realizar un análisis comparativo entre todos ellos. Adicionalmente, se pueden complementar los resultados de diferentes repositorios y centralizarlos en la representación de recursos diseñada en este trabajo en función de: la interoperabilidad, migración, intercambio de recursos, calidad de los metadatos, etc.
- El enfoque Model-Driven provee de muchas herramientas alternativas propietarias o con licencias abiertas, situación que abre un espectro amplio de posibilidades de desarrollo de un marco de referencia como el que se propuso en este estudio, por ello, se puede realizar un estudio de las diferentes alternativas de herramientas de desarrollo de aplicaciones web siempre bajo el enfoque MDE.
- Implementar otras funcionalidades del dominio LIS que permitan tener un criterio más certero del alcance real de WebRatio, así como también estudiar la integración de dichas funcionalidades (aplicaciones) con los estándares como BPM e IFML que se encuentran en WebRatio. De igual manera, se deben

estudiar otras propuestas similares a WebRatio, para terminar de desarrollar las funcionalidades deseadas en el marco de referencia implementado en esta investigación.

- Un área de estudio nueva en el enfoque Model-Driven es el desarrollo de transformaciones bidireccionales, que pueden ser de utilidad para ampliar la investigación presentada. Establecer una relación desde el modelo propuesto a las diferentes plataformas de software de los repositorios, generará avances sobre los principios de interoperabilidad y preservación de los recursos en dichos sistemas de información. Aunque esa bidireccionalidad esté sujeta a las personalizaciones de los sistemas de representación de recursos a los RI, situación que la hace más difícil y compleja.
- El esquema propuesto por basarse sobre un enfoque Model-Driven, resulta muy amplio y abarcativo; por ello, se decidió dejar a futuro el desarrollo de componentes de software que fortalecerán tal iniciativa, por ejemplo: realizar una aplicación web que muestre de forma embebida todo el marco de referencia propuesto, gracias a que todas las fases se pueden convertir en plugins para ser reconocidos por una aplicación web, metamodelos para interpretar los AIP o los diferentes archivos que provienen de Dspace-SEDICI, generación de ontologías a partir de la información recuperada para uno o varios repositorios importados, relación de un metamodelo de WebRatio y la propuesta presentada, desarrollar otros módulos funcionales de los repositorios institucionales, desarrollar aplicaciones para otros dispositivos, etc.
- Existen pruebas dirigidas por modelos que permiten validar y verificar el software desarrollado en esta propuesta, pero no hay mucho avance al respecto. Esas pruebas deben estar focalizadas en el proyecto interno (proyecto Model-Driven) que debe tener la participación de los usuarios finales para validar y verificar que se está desarrollando el software correcto. Existen otros tipos de pruebas que enfocadas al proyecto externo, que serán los dueños del negocio y expertos del dominio quienes validan y verifican las funcionalidades obtenidas.
- El desarrollo de la cuarta fase (secciones 5.2.4., 5.3.4. & 5.4.4.), da cuenta de la necesidad de diseñar metamodelos que se ajusten a la sintaxis de archivos tan

variados como los estudiados, siete en total: recursos exportados por DSpace, comunidades, colecciones, autores, instituciones, revistas y números de revistas. El principal inconveniente está en la no existencia de estándares que regularicen la parte sintáctica y semántica de ellos, por ello, se abre la posibilidad de desarrollar otras alternativas para solucionar este problema.

# Referencias


Acerbis, R., Bongio, A., Brambilla, M., Butti, S., Ceri, S., & Fraternali, P. (2008). Web Applications Design and Development with WebML and WebRatio 5.0. In R. F. Paige & B. Meyer (Eds.), *Objects, Components, Models and Patterns* (pp. 392–411). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-69824-1_22

Agenjo, X., & Hernández, F. (2010). Tendencias internacionales en el desarrollo funcional de la recuperación de la información: Linked Open Data (LOD). Presented at the X Workshop Rebiun sobre proyectos digitales: diez años de proyectos digitales: cambian las bibliotecas, cambian los profesionales. Valencia, 7 y 8 de octubre de 2010. Retrieved from http://riunet.upv.es/handle/10251/8665

Ahrefs. (2014). Ahrefs Site Explorer. Retrieved March 13, 2014, from http://ahrefs.com/

Altenhöner, R. (2006). Data for the Future: the German Project "Co-operative Development of a Long-term Digital Information Archive" (kopal) [Journal article (Print/Paginated)]. Retrieved November 11, 2013, from http://eprints.rclis.org/9199/

Arms, W. Y. (2001). *Digital Libraries*. MIT Press.

ASIS&T. (2014a). DC-Library Application Profile (DC-Lib). Retrieved July 9, 2014, from http://dublincore.org/documents/library-application-profile/

ASIS&T. (2014b). DCMI Abstract Model. Retrieved July 9, 2014, from http://dublincore.org/documents/abstract-model/

Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice* (2nd ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Bawden, D., & Rowlands, I. (1999a). Digital libraries: assumptions and concepts. Retrieved September 13, 2012, from http://discovery.ucl.ac.uk/166226/

Bawden, D., & Rowlands, I. (1999b). *Understanding Digital Libraries: Towards a Conceptual Framework*. British Library Research and Innovation Centre.

Björk, B.-C., & Solomon, D. (2012). Open access versus subscription journals: a comparison of scientific impact. *BMC Medicine*, *10*(1), 73. doi:10.1186/1741-7015-10-73

Borgman, C. L. (1999). What Are Digital Libraries? Competing Visions. *Information Processing & Management*, *35*(3), 227–243.

Brambilla, M., Cabot, J., & Wimmer, M. (2012). *Model-Driven Software Engineering in Practice*. Morgan & Claypool.

Brambilla, M., Comai, S., Fraternali, P., & Matera, M. (2008). Designing Web Applications with Webml and Webratio. In G. Rossi, O. Pastor, D. Schwabe, & L. Olsina (Eds.), *Web Engineering: Modelling and Implementing Web Applications* (pp. 221–261). Springer London. Retrieved from http://link.springer.com/chapter/10.1007/978-1-84628-923-1_9

Brambilla, M., & Fraternali, P. (2013). Large-scale Model-Driven Engineering of web user interaction: The WebML and WebRatio experience. *Science of Computer Programming*. doi:10.1016/j.scico.2013.03.010

Brindley, G., Muir, A., & Probets, S. (2004). Provision of digital preservation metadata: a role for ONIX? *Program: Electronic Library and Information Systems*, *38*(4),

240–250. doi:10.1108/00330330410699883
Brooks, F. (1987). No Silver Bullet Essence and Accidents of Software Engineering. *Computer*, *20*(4), 10–19. doi:10.1109/MC.1987.1663532
Bustos-González, A., & Fernández-Porcel, A. (2008). Directrices para la creación de repositorios institucionales en universidades y organizaciones de educación superior. *Universidad del Rosario*. Retrieved from http://repository.urosario.edu.co//handle/10336/223
Candela, L., Castelli, D., Ioannidis, Y., Ross, S., Thanos, C., Pagano, P., … Schuldt, H. (2007). Setting the Foundations of Digital Libraries. *D-Lib*, *13*(3/4). Retrieved from http://www.dlib.org/dlib/march07/castelli/03castelli.html
Castells, M. (2009). *The Rise of the Network Society: The Information Age: Economy, Society, and Culture Volume I* (2nd Edition with a New Preface.). Wiley-Blackwell.
CCSDS. (2012). *Reference Model for an Open Archival Information System (OAIS):ISO 14721*.
Ceri, S., Fraternali, P., & Bongio, A. (2000). Web Modeling Language (WebML): a modeling language for designing Web sites. *Computer Networks*, *33*(1–6), 137–157. doi:10.1016/S1389-1286(00)00040-2
Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., & Matera, M. (2003). *Designing Data-Intensive Web Applications*. Morgan Kaufmann.
Ceri, S., Fraternali, P., & Matera, M. (2002). Conceptual modeling of data-intensive Web applications. *IEEE Internet Computing*, *6*(4), 20–30. doi:10.1109/MIC.2002.1020321
Chan, L. M., & Zeng, M. L. (2006). Metadata Interoperability and Standardization - A Study of Methodology, Part I. *D-Lib Magazine*, *12*(6), 3–.
Chowdhury, G. G., & Chowdhury, S. (1999). Digital library research: major issues and trends. *Journal of Documentation*, *55*(4), 409–448. doi:10.1108/EUM0000000007154
Chua, A. Y. K., & Yang, C. C. (2008). The shift towards multi-disciplinarity in information science. *Journal of the American Society for Information Science and Technology*, *59*(13), 2156–2170. doi:10.1002/asi.20929
Cold, S. J. (2006). Using Really Simple Syndication (RSS) to Enhance Student Research. *SIGITE Newsl.*, *3*(1), 6–9. doi:10.1145/1113378.1113379
CRM. (2014). The CIDOC CRM. Retrieved March 9, 2014, from http://www.cidoc-crm.org/
Crow, R. (2002). The Case for Institutional Repositories : A SPARC Position Paper. *The Scholarly Publishing & Academic Resources Coalition*, 1–37.
CSIC. (2014). Ranking Web of Repositories. Retrieved July 8, 2014, from http://repositories.webometrics.info/
DAMS. (2014). University of Texas at Austin Digital Asset Management System. Retrieved March 9, 2014, from http://www.lib.utexas.edu/dams/
DCMI. (2014). Dublin Core Metadata Element Set, Version 1.1. Retrieved July 13, 2014, from http://www.dublincore.org/documents/dces/
De Giusti, M., Oviedo, N., Lira, A., Sobrado, A., Martinez, J., & Pinto, A. (2011). SEDICI – Desafíos y experiencias en la vida de un repositorio digital. *RENATA*, *1*(2), 16–33.

De Giusti, M. R., Lira, A. J., Villarreal, G. L., & Texier, J. (2012). Las actividades y el planeamiento de la preservación en un repositorio institucional. Presented at the BIREDIAL - Conferencia Internacional Acceso Abierto, Comunicación Científica y Preservación Digital. Retrieved from http://hdl.handle.net/10915/26045

De Giusti, M. R., Villarreal, G. L., Terruzzi, F. A., Oviedo, N., & Lira, A. J. (2013). Interoperabilidad entre el Repositorio Institucional y servicios en línea en la Universidad Nacional de La Plata. Presented at the PKP International Scholarly Publishing Conferences 2013 (Mexico). Retrieved from http://hdl.handle.net/10915/27406

DeMarco, T. (1979). Structured Analysis and System Specification (pp. 409–424). Upper Saddle River, NJ, USA: Yourdon Press. Retrieved from http://dl.acm.org/citation.cfm?id=1241515.1241539

De Souza-Silva, J., Cheaz Peláez, J., & Calderón Romero, J. (2001). *La cuestión institucional, de la vulnerabilidad a la sostenibilidad institucional en el contexto del Cambio de Epoca*. Costa Rica: Servicio Internacional para la Investigación Agrícola Nacional - ISNAR.

DLI. (1998). Digital Libraries Initiative - Phase II. Retrieved July 9, 2012, from http://www.nsf.gov/pubs/1998/nsf9863/nsf9863.htm

DL.org. (2014). DL.org Reference Model. Retrieved March 9, 2014, from http://www.dlorg.eu/index.php/outcomes/reference-modeloutcomes/reference-model

Dodero, J. M., Palomo-Duarte, M., & Karampiperis, P. (2012). *Metadata and Semantics Research: 6th Research Conference, MTSR 2012*. Cadiz, Spain: Springer Publishing Company, Incorporated.

DSpace. (2014). DSpace. Retrieved March 12, 2014, from http://www.dspace.org/

Duval, E., Hodgins, W., Sutton, S., & Weibel, S. L. (2002). Metadata Principles and Practicalities. *D-Lib Magazine*, *8*(4). doi:10.1045/april2002-weibel

Eclipse. (2014a). Eclipse - The Eclipse Foundation open source community website. Retrieved April 30, 2014, from http://www.eclipse.org/

Eclipse. (2014b). Graphical Modeling Framework. Retrieved April 30, 2014, from http://www.eclipse.org/modeling/gmp/

EPrints. (2014). EPrints - Digital Repository Software. Retrieved March 9, 2014, from http://www.eprints.org/

EU. (2014). Europeana Data Model. Retrieved March 8, 2014, from http://pro.europeana.eu/edm-documentation

Fedora. (2014). Fedora Repository. Retrieved March 9, 2014, from http://www.fedora-commons.org/

Flores, G., & Sánchez, N. (2007). Los repositorios institucionales: análisis de la situación internacional y principios generales para Cuba. *ACIMED*, *16*(6), 0–0.

FOSS. (2014). Free Open Source Software. Retrieved May 12, 2014, from http://freeopensourcesoftware.org

Fowler, M. (2010). *Domain Specific Languages* (1st ed.). Addison-Wesley Professional.

Fox, E. A., Gonçalves, M. A., & Shen, R. (2012). *Theoretical Foundations for Digital Libraries. The 5S Approach*. North Carolina: Morgan & Claypool Publishers.

FRBR. (2009). Functional Requirements for Bibliographic Records - IFLA. Retrieved

from http://www.ifla.org/files/cataloguing/frbr/frbr.pdf
Gerber, A., & Raymond, K. (2003). MOF to EMF: There and Back Again. In *Proceedings of the 2003 OOPSLA Workshop on Eclipse Technology eXchange* (pp. 60–64). New York, NY, USA: ACM. doi:10.1145/965660.965673
GitHub. (2014a). GitHub · Build software better, together. Retrieved July 30, 2014, from https://github.com/
GitHub. (2014b). Propuesta para la tesis - GitHub. Retrieved July 30, 2014, from https://github.com/dantexier/Tesis
Gonçalves, M. A., Fox, E. A., Watson, L. T., & Kipp, N. A. (2004). Streams, structures, spaces, scenarios, societies (5s): A formal model for digital libraries. *ACM Trans. Inf. Syst.*, *22*(2), 270–312. doi:10.1145/984321.984325
Greenberg, J. (2005). Understanding Metadata and Metadata Schemes. *Cataloging & Classification Quarterly*, *40*(3-4), 17–36. doi:10.1300/J104v40n03_02
Griffin, S. M. (1998). Taking the Initiative for Digital Libraries. *Electronic Library*, *16*(1), 24–27.
Gronback, R. C. (2009). *Eclipse Modeling Project: A Domain-Specific Language* (1 edition.). Upper Saddle River, NJ: Addison-Wesley Professional.
Guerra, E., de Lara, J., Malizia, A., & Díaz, P. (2009). Supporting user-oriented analysis for multi-view domain-specific visual languages. *Information and Software Technology*, *51*(4), 769–784. doi:10.1016/j.infsof.2008.09.005
Guerra, E., Lara, J., & Malizia, A. (2007). Model driven development of digital libraries-Validation, analysis and code generation. In *Webist 2007 - 3rd International Conference on Web Information Systems and Technologies, Proceedings* (Vol. WIA, pp. 35–42).
Guo, L. (2010). On construction of digital libraries in universities. In *2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)* (Vol. 1, pp. 452 –456). doi:10.1109/ICCSIT.2010.5564750
Heery, R. (1996). Review of metadata formats. *Program: Electronic Library and Information Systems*, *30*(4), 345–373. doi:10.1108/eb047236
Hurtado, J. (2008). *Cómo formular objetivos de investigación* (2da ed.). Caracas, Venezuela: Fundacion Sypal.
IFLA. (2014a). Functional Requirements for Authority Data. Retrieved March 8, 2014, from http://www.ifla.org/publications/functional-requirements-for-authority-data
IFLA. (2014b). Functional Requirements for Subject Authority Data. Retrieved March 8, 2014, from http://www.ifla.org/node/5849
ISO. (2014). ISO/IEC 12207:2008 - Systems and software engineering -- Software life cycle processes. Retrieved September 16, 2014, from http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43447
IU. (2014). Variations: The Indiana University Digital Music Library. Retrieved March 9, 2014, from http://variations.indiana.edu/index.html
Jeevan, V. K. J. (2004). Digital library development: identifying sources of content for developing countries with special reference to India. *The International Information & Library Review*, *36*(3), 185–197. doi:10.1016/j.iilr.2003.10.005
Jones, R. (2013). SWORD 2.0 Profile. Retrieved October 29, 2013, from http://swordapp.github.io/SWORDv2-Profile/SWORDProfile.html

Kleppe, A., Warmer, J., & Bast, W. (2003). *MDA Explained: The Model Driven Architecture(TM): Practice and Promise*. Addison-Wesley Professional. Retrieved from http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/032119442X

Kochtanek, T. R., & Hein, K. K. (1999). Delphi study of digital libraries. *Information Processing & Management*, *35*(3), 245–254. doi:10.1016/S0306-4573(98)00060-0

Kökörvcený, M., & Bodnárová, A. (2010). Comparison of digital libraries systems. In *Proceedings of the 9th WSEAS international conference on Data networks, communications, computers* (pp. 97–100). Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS). Retrieved from http://dl.acm.org/citation.cfm?id=1948805.1948823

Kroeger, A. (2013). The Road to BIBFRAME: The Evolution of the Idea of Bibliographic Transition into a Post-MARC Future. *Cataloging & Classification Quarterly*, *51*(8), 873–890. doi:10.1080/01639374.2013.823584

Lagoze, C., & Hunter, J. (2006). The ABC Ontology and Model. *Journal of Digital Information*, *2*(2). Retrieved from http://journals.tdl.org/jodi/index.php/jodi/article/view/44

LC. (2014). BIBFRAME - Bibliographic Framework Initiative (Library of Congress). Retrieved March 8, 2014, from http://www.loc.gov/bibframe/

Lesk, M. (1997). *Practical Digital Libraries: Books, Bytes, and Bucks*. Morgan Kaufmann.

Liddle, S. W. (2011). Model-Driven Software Development. In D. W. Embley & B. Thalheim (Eds.), *Handbook of Conceptual Modeling* (pp. 17–54). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-15865-0_2

Liew, C. L. (2009). Digital library research 1997-2007: Organisational and people issues. *Journal of Documentation*, *65*(2), 245–266. doi:10.1108/00220410910937606

López-Gúzman, C. (2005). Los Repositorios de Objetos de Aprendizaje como soporte a un entorno e-learning [Trabajo de Grado]. Retrieved March 6, 2014, from http://gredos.usal.es/jspui/handle/10366/56649

Lynch, C. A. (2003). Institutional Repositories: Essential Infrastructure for Scholarship in the Digital Age. Retrieved October 28, 2013, from http://www.arl.org/resources/pubs/br/br226/br226ir.shtml

Madalli, D. P., Barve, S., & Amin, S. (2012). Digital Preservation in Open-Source Digital Library Software. *The Journal of Academic Librarianship*, *38*(3), 161–164. doi:10.1016/j.acalib.2012.02.004

MADS. (2014). Metadata Authority Description Schema (MADS) - (Library of Congress). Retrieved July 11, 2014, from http://www.loc.gov/standards/mads/

Majestic. (2014). Majestic SEO : Backlink Checker & Site Explorer. Retrieved March 13, 2014, from http://www.majesticseo.com/

Malizia, A., Bottoni, P., & Levialdi, S. (2010). Generating Collaborative Systems for Digital Libraries: a Model-Driven Approach. *Information Technology & Libraries*, *29*. Retrieved from http://www.ala.org/lita/ital/files/29/4/malizia.pdf

Malizia, A., Bottoni, P., Levialdi, S., & Astorga-Paliza, F. (2007). A Cooperative-

Relational Approach to Digital Libraries. In L. Kovács, N. Fuhr, & C. Meghini (Eds.), *Research and Advanced Technology for Digital Libraries* (pp. 75–86). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-74851-9_7

MARC. (2014). MARC STANDARDS (Network Development and MARC Standards Office, Library of Congress). Retrieved March 11, 2014, from http://www.loc.gov/marc/

Martínez-Tamayo, A. M., & Valdez, J. C. (2009). *Indicación y Clasificación en Bibliotecas* (Primera edición.). Argentina: Alfagrama.

McNicol, S. (2003). LIS the interdisciplinary research landscape. *Journal of Librarianship and Information Science*, *35*(1), 23–30. doi:10.1177/096100060303500103

Méndez, E. (2003a). La descripción de documentos electrónicos a través de metadatos : una visión para la Archivística desde la nueva e- Administración [Journal Article (Print/Paginated)]. Retrieved June 5, 2013, from http://eprints.rclis.org/12684/

Méndez, E. (2003b). Tratamiento de los objetos de información en los archivos : retos y estándares para la descripción basada en metadatos [Book Chapter]. Retrieved October 13, 2013, from http://eprints.rclis.org/handle/10760/12691#.UAAZFuEzfgM

METS. (2014). Metadata Encoding and Transmission Standard (METS) Official Web Site. Retrieved March 14, 2014, from http://www.loc.gov/standards/mets/

Mischo, W. H., Norman, M. A., Shelburne, W. A., & Schlembach, M. C. (2007). The Growth of Electronic Journals in Libraries. *Science & Technology Libraries*, *26*(3-4), 29–59. doi:10.1300/J122v26n03_04

MODS. (2014). Metadata Object Description Schema: MODS (Library of Congress). Retrieved March 13, 2014, from http://www.loc.gov/standards/mods/

Navarro, A., Cristóbal, J., Fernández-Chamizo, C., & Fernández-Valmayor, A. (2011). Architecture of a multiplatform virtual campus. *Software: Practice and Experience*. doi:10.1002/spe.1130

Nguyen, S., & Chowdhury, G. (2011). Digital Library Research (1990-2010): A Knowledge Map of Core Topics and Subtopics. In C. Xing, F. Crestani, & A. Rauber (Eds.), *Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation* (Vol. 7008, pp. 367–371). Springer Berlin / Heidelberg. Retrieved from http://www.springerlink.com/content/21h17m2nh10krl1w/abstract/

NISO. (2014). The OpenURL Framework (ANSI/NISO Z39.88-2004). Retrieved March 9, 2014, from http://www.niso.org/apps/group_public/project/details.php?project_id=82

OAI. (2014). Open Archives Initiative Protocol for Metadata Harvesting. Retrieved March 11, 2014, from http://www.openarchives.org/pmh/

OMG. (2003). *MDA Guide Version 1.0.1* (p. 62).

OMG. (2014). XMI. Retrieved May 8, 2014, from http://www.omg.org/spec/XMI/

OpenDOAR. (2014). OpenDOAR - Home Page - Directory of Open Access Repositories. Retrieved July 21, 2014, from http://www.opendoar.org/

OpenRefine. (2014). OpenRefine. Retrieved May 28, 2014, from http://openrefine.org/

OpenSearch. (2014). OpenSearch Specifications. Retrieved March 29, 2014, from http://www.opensearch.org/Specifications/OpenSearch/1.1

Paganelli, F., & Pettenati, M. C. (2006). A Model-driven Method for the Design and Deployment of Web-based Document Management Systems. *Journal of Digital Information*, *6*(3). Retrieved from http://journals.tdl.org/jodi/article/view/67

Piwowar, H. A., Day, R. S., & Fridsma, D. B. (2007). Sharing Detailed Research Data Is Associated with Increased Citation Rate. *PLoS ONE*, *2*(3), e308. doi:10.1371/journal.pone.0000308

Pons, C., Giandini, R., & Pérez, G. (2010). *Desarrollo de Software Dirigido por Modelos*. La Plata, Argentina: Mc Graw Hill.

PREMIS. (2008). *PREMIS Data Dictionary for Preservation Metadata -- version 2.0*. Retrieved from http://www.loc.gov/standards/premis/v2/premis-2-0.pdf

PREMIS. (2014). PREMIS: Preservation Metadata Maintenance Activity (Library of Congress). Retrieved March 13, 2014, from http://www.loc.gov/standards/premis/

Prensky, M. (2001). Digital Natives, Digital Immigrants. *MCB University Press*, *9*(5). Retrieved from http://www.marcprensky.com/writing/Prensky%20-%20Digital%20Natives,%20Digital%20Immigrants%20-%20Part1.pdf

Pressman, R. (2002). *Ingeniería del Software* (Quinta edición.). Mc Graw Hill.

Pyrounakis, G., & Nikolaidou, M. (2009). Comparing Open Source Digital Library Software. *Handbook of Research on Digital Libraries: Design, Development, and Impact*, 51–60. doi:doi:10.4018/978-1-59904-879-6.ch006

Riley, J. (2010). Seeing Standards. Retrieved December 2, 2013, from http://www.dlib.indiana.edu/~jenlrile/metadatamap/

ROAR. (2014). Registry of Open Access Repositories (ROAR). Retrieved July 21, 2014, from http://roar.eprints.org/

RSS. (2014). RSS 2.0 Specification. Retrieved March 11, 2014, from http://www.rssboard.org/rss-specification

Rusch-Feja, D. (2002). The Open Archives Initiative and the OAI Protocol for Metadata Harvesting: rapidly forming a new tier in the scholarly communication infrastructure. *Learned Publishing*, *15*(3), 179–186. doi:10.1087/095315102320140464

SEDICI. (2014). SEDICI - Repositorio de la Universidad Nacional de La Plata. Retrieved July 14, 2014, from http://sedici.unlp.edu.ar/

Sharon, T., & Frank, A. (2001). Digital libraries on the Internet. *66th IFLA Council and General Conference*. Retrieved from http://archive.ifla.org/IV/ifla66/papers/029-142e.htm

Singh, S. K., Witt, M., & Dorothea, S. (2010). A Comparative Analysis of Institutional Repository Software. Presented at the Fifth International Conference on Open Repositories, Madrid. Retrieved from http://www.or10.es/Resources/documentos/GSabstracts/AComparativeAnalysisInstitutionalRepositorySoftware.pdf

SJC. (2014). Scimago Journal & Country Rank. Retrieved July 28, 2014, from http://www.scimagojr.com/

Steinberg, D., Budinsky, F., Merks, E., & Paternostro, M. (2008). *EMF: Eclipse Modeling Framework* (Second.). Pearson Education.

Suber, P. (2012). Ensuring open access for publicly funded research. *BMJ : British Medical Journal*, *345*. doi:10.1136/bmj.e5184

Tansley, R., Bass, M., & Smith, M. (2003). DSpace as an Open Archival Information System: Current Status and Future Directions. In T. Koch & I. T. Sølvberg (Eds.), *Research and Advanced Technology for Digital Libraries* (Vol. 2769, pp. 446–460). Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://www.springerlink.com/content/hdepd4443hl00k4k/

Texier, J. (2013). Los repositorios institucionales y las bibliotecas digitales: una somera revisión bibliográfica y su relación en la educación superior (p. 9). Presented at the 11th Latin American and Caribbean Conference for Engineering and Technology - 2013, Cancun, Mexico: LACCEI. Retrieved from http://eprints.rclis.org/19925/

Texier, J., & De Giusti, M. (2014). Elements of Resource Representation in Institutional Repositories: a Bibliographic Review. *Journal of Information and Organizational Sciences*, *38*(1). Retrieved from https://jios.foi.hr/index.php/jios/article/view/816

Texier, J., De Giusti, M., & Gordillo, S. (2014). Model-driven software development in the institutional repositories. *DYNA*, *81*(184), 7.

Texier, J., De Giusti, M., Lira, A., & Villarreal, G. (2014). La visualización de autores en un Repositorio Institucional a través del enfoque Model Driven con WebRatio. *Revista Investigación Bibliotecológica*. Retrieved from http://cuib.unam.mx/revistaCuib.html

Texier, J., De Giusti, M. R., Lira, A. J., Oviedo, N., & Villarreal, G. L. (2013). DSpace como herramienta para un repositorio de documentos administrativos en la Universidad Nacional Experimental del Táchira. *Revista Interamericana de Bibliotecología*, *36*(2), 109–124.

Texier, J., De Giusti, M. R., Oviedo, N., Lira, A. J., & Villarreal, G. L. (2013). La representación de recursos en los repositorios institucionales. El caso de estudio: SEDICI. Presented at the III Conferencia de Bibliotecas y Repositorios Digitales de América Latina (BIREDIAL) y VIII Simposio Internacional de Bibliotecas Digitales (SIBD) (Costa Rica, 2013). Retrieved from http://hdl.handle.net/10915/30111

Texier, J., Giusti, M., Lira, A., & Villarreal, G. (2014). La incorporación del registro de obras y autores de una institución: estudio de caso SEDICI-UNLP-Scopus. IV Conferência Internacional sobre Bibliotecas e Repositórios Digitais (BIREDIAL 2014). IX Simpósio Internacional de Bibliotecas Digitais (SIBD 2014).

Torres-Salinas, D., Robinson-García, N., & Cabezas-Clavijo, A. (2012). Compartir los datos de investigación en ciencia: introducción al data sharing. *Profesional de La Informacion*, *21*(2), 173–184.

Tramullas, J. (2002). Propuestas de concepto y definición de la biblioteca digital, 11–20.

Tramullas, J. (2007). Bibliotecas digitales. Presented at the VI Seminario de Centros de Documentación Ambiental y Espacios Naturales Protegidos. Retrieved from http://eprints.rclis.org/handle/10760/11304#.T6qpEhQzfgN

Tramullas, J., & Garrido, P. (2006). Software libre para repositorios institucionales: propuestas para un modelo de evaluación de prestaciones. *El Profesional de La Información*, *15*(3), 171–181.

Tzoc, E. (2013). A Mobile Interface for DSpace. *D-Lib Magazine*, *19*(3/4). doi:10.1045/march2013-tzoc

UIUC. (2014). OAI Registry at UIUC. Retrieved July 21, 2013, from http://gita.grainger.uiuc.edu/registry/

UKOLN. (2014). UKOLN | Metadata. Retrieved July 9, 2014, from http://www.ukoln.ac.uk/metadata/

Van de Sompel, H., Payette, S., Erickson, J., Lagoze, C., & Warner, S. (2004). Rethinking Scholarly Communication. *D-Lib Magazine*, *10*(9). doi:10.1045/september2004-vandesompel

Vogel, L. (2014). Eclipse Modeling Framework (EMF) - Tutorial. Retrieved April 30, 2014, from http://www.vogella.com/tutorials/EclipseEMF/article.html

W3C. (2014a). Extensible Markup Language (XML). Retrieved May 8, 2014, from http://www.w3.org/XML/

W3C. (2014b). XML Schema Definition. Retrieved May 8, 2014, from http://www.w3.org/TR/xmlschema11-1/

Waters, D. (1998). What Are Digital Libraries? *CLIR Issues*, *4*. Retrieved from http://www.clir.org/pubs/issues/issues04.html

WebRatio. (2013). WebRatio. Retrieved January 20, 2014, from http://www.webratio.com/portal/content/en/home

Weng, C., & Mi, J. (2006). Towards accessibility to digital cultural materials: a FRBRized approach. *OCLC Systems & Services*, *22*(3), 217–232. doi:10.1108/10650750610686766

Whittle, J., Clark, T., & Kühne, T. (Eds.). (2011). *Model Driven Engineering Languages and Systems*. Wellington, New Zealand,: MODELS 2011. Retrieved from http://www.springerlink.com/content/978-3-642-24484-1#section=971719&page=1

Witten, I. H., Bainbridge, D., & Nichols, D. M. (2003). *How to Build a Digital Library*. Morgan Kaufmann.

Xia, J., & Opperman, D. B. (2010). Current Trends in Institutional Repositories for Institutions Offering Master's and Baccalaureate Degrees. *Serials Review*, *36*(1), 10–18. doi:10.1016/j.serrev.2009.10.003

Yang, S., Pomerantz, J., Wildemuth, B. M., & Fox, E. A. (2006). Curriculum development for digital libraries. In *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries, 2006. JCDL '06* (pp. 175 –184). doi:10.1145/1141753.1141787

Yates, J. (1989). *Control through communication*. Baltimore: Johns Hopkins University Press.

Zavalina, O. L. (2012). Subject Access: Conceptual Models, Functional Requirements, and Empirical Data. *Journal of Library Metadata*, *12*(2-3), 140–163. doi:10.1080/19386389.2012.699829

# Apéndices

## Apéndice A

Código fuente – Fase 1

El código descrito en esta sección corresponde a la fase de la construcción del DSL que soporta las recomendaciones de la norma ISO 14721. Todo el código fuente de esta fase se encuentra en un proyecto GitHub llamado Tesis/Fase1 (GitHub, 2014b). La estructura principal de archivos del proyecto GMF es:

```
/Norma/                          ==>  proyecto principal donde se encuentran los metamodelos
....../SimpleClass.ecore         ==>  metamodelo base (Tabla A-1)
....../SimpleClass.ecorediag
....../SimpleClass.genmodel
....../SimpleClass.gmfgen
....../SimpleClass.gmfgraph
....../SimpleClass.gmfmap
....../SimpleClass.gmftool
....../SimpleClass.trace
/Norma.diagram/
/Norma.edit/
/Norma.editor/
/Norma.tests/
/runtime-Norma/                  ==>  proyecto donde se almacena el modelo diseñado
....../Modelo.XMI                ==>  Tabla A-2
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore">
  <ecore:EPackage name="SimpleClass" nsURI="sedici.Simple" nsPrefix="SimpleClass">
    <eClassifiers xsi:type="ecore:EClass" name="Classifier">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" ordered="false"
          lowerBound="1" eType="#/1/String"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="Class" eSuperTypes="#/0/Classifier">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="is_persistent"
ordered="false"
          lowerBound="1" eType="#/1/Boolean"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="parent" ordered="false"
          eType="#/0/Class"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="attrs" ordered="false"
          upperBound="-1" eType="#/0/Attribute" containment="true"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="PrimitiveDataType"
eSuperTypes="#/0/Classifier"/>
    <eClassifiers xsi:type="ecore:EClass" name="Association">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" ordered="false"
          lowerBound="1" eType="#/1/String"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="src" ordered="false"
          lowerBound="1" eType="#/0/Class"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="dest" ordered="false"
          lowerBound="1" eType="#/0/Class"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="Attribute">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" ordered="false"
          lowerBound="1" eType="#/1/String"/>
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="is_primary"
```

```
ordered="false"
          lowerBound="1" eType="#/1/Boolean"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="type" ordered="false"
          lowerBound="1" eType="#/0/Classifier"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EEnum" name="EA">
      <eLiterals name="Author"/>
      <eLiterals name="Institution" value="1"/>
      <eLiterals name="Journal" value="2"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EEnum" name="Vocabulary">
      <eLiterals name="Normal"/>
      <eLiterals name="Language" value="1"/>
      <eLiterals name="Eurovocs" value="2"/>
      <eLiterals name="Decs" value="3"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="Schema" eSuperTypes="#/0/Classifier">
      <eStructuralFeatures xsi:type="ecore:EReference" name="clases" upperBound="-1"
          eType="#/0/Class" containment="true"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="relaciones" upperBound="-
1"
          eType="#/0/Association" containment="true"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="datos" upperBound="-1"
          eType="#/0/PrimitiveDataType" containment="true"/>
    </eClassifiers>
  </ecore:EPackage>
  <ecore:EPackage name="PrimitiveTypes" nsURI="sedici.PT" nsPrefix="PrimitiveTypes">
    <eClassifiers xsi:type="ecore:EDataType" name="String"
instanceClassName="java.lang.String"/>
    <eClassifiers xsi:type="ecore:EEnum" name="Entity" instanceTypeName="EA"/>
    <eClassifiers xsi:type="ecore:EDataType" name="Boolean"
instanceClassName="java.lang.Boolean"/>
    <eClassifiers xsi:type="ecore:EDataType" name="Int"
instanceClassName="java.lang.Integer"/>
  </ecore:EPackage>
</xmi:XMI>
```

*Tabla A-1. Código del metamodelo SimpleClass.ecore*

```
<?xml version="1.0" encoding="UTF-8"?>
<SimpleClass:Schema xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:SimpleClass="sedici.Simple" name="Ejemplo">
  <clases name="CI" is_persistent="true">
    <attrs name="ID" is_primary="true" type="//@datos.1"/>
    <attrs name="Titulo" is_primary="false" type="//@datos.0"/>
  </clases>
  <clases name="Metadatos" is_persistent="true">
    <attrs name="ID" is_primary="true" type="//@datos.1"/>
    <attrs name="Valor" is_primary="false" type="//@datos.0"/>
    <attrs name="Tipo" is_primary="false" type="//@datos.0"/>
    <attrs name="Vocabulario" is_primary="false" type="//@datos.0"/>
  </clases>
  <clases name="EntidadAbstracta" is_persistent="true">
    <attrs name="nombre" is_primary="true" type="//@datos.0"/>
    <attrs name="tipo" is_primary="true" type="//@datos.0"/>
  </clases>
  <relaciones name="R1" src="//@clases.1" dest="//@clases.0"/>
  <relaciones name="R2" src="//@clases.1" dest="//@clases.2"/>
  <datos name="String"/>
  <datos name="Int"/>
  <datos name="EA"/>
  <datos name="Normal"/>
</SimpleClass:Schema>
```

*Tabla A-2. Código de ejemplo de un modelo ajustado a SimpleClass.ecore*

## Código fuente – Fase 2

El código descrito en esta sección corresponde a la transformación M2M en ATL. El código fuente de esta fase se encuentra en un proyecto GitHub con el título Tesis/Fase2 (GitHub, 2014b). La estructura principal de archivos del proyecto ATL es:

```
/Tesis/                              ==> proyecto principal con los metamodelos
      /Metamodelos/
      ....../SimpleClass.ecore       ==> metamodelo base de entrada
      ....../SimpleRDBMS.ecore       ==> metamodelo base de salida (Tabla A-3)
      ....../SimpleRDBMS.genmodel    ==> Tabla A-4
      /Modelos/
      ....../norma.XMI               ==> modelo de entrada (Tabla A-5)
      ....../basededatos.XMI         ==> modelo de salida (Tabla A-6)
      /Transformacion/
      ....../M2M.atl                 ==> Tabla A-7
/Tesis.edit/
/Tesis.editor/
/Tesis.tests/
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore">
  <ecore:EPackage name="PrimitiveTypes" nsURI="sedici.PT" nsPrefix="PrimitiveTypes">
    <eClassifiers xsi:type="ecore:EDataType" name="String"
instanceClassName="java.lang.String"/>
  </ecore:EPackage>
  <ecore:EPackage name="SimpleRDBMS" nsURI="sedici.SimpleRDBMS" nsPrefix="SimpleRDBMS">
    <eClassifiers xsi:type="ecore:EClass" name="Table">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" ordered="false"
          lowerBound="1" eType="#/0/String"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="fkeys" ordered="false"
          upperBound="-1" eType="#/1/FKey" containment="true"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="pkey" ordered="false"
          upperBound="-1" eType="#/1/Column"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="cols" ordered="false"
          upperBound="-1" eType="#/1/Column" containment="true"/>
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="tipo" eType="#/0/String"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="Column">
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" ordered="false"
          lowerBound="1" eType="#/0/String"/>
      <eStructuralFeatures xsi:type="ecore:EAttribute" name="type" ordered="false"
          lowerBound="1" eType="#/0/String"/>
    </eClassifiers>
    <eClassifiers xsi:type="ecore:EClass" name="FKey">
      <eStructuralFeatures xsi:type="ecore:EReference" name="references"
ordered="false"
          lowerBound="1" eType="#/1/Table"/>
      <eStructuralFeatures xsi:type="ecore:EReference" name="cols" ordered="false"
          upperBound="-1" eType="#/1/Column"/>
    </eClassifiers>
  </ecore:EPackage>
</xmi:XMI>
```

*Tabla A-3. Código del metamodelo SimpleRDBMS.ecore*

```
<?xml version="1.0" encoding="UTF-8"?>
<genmodel:GenModel xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
```

```
    xmlns:genmodel="http://www.eclipse.org/emf/2002/GenModel"
modelDirectory="/Tesis/src" modelPluginID="Tesis" modelName="SimpleRDBMS"
    rootExtendsClass="org.eclipse.emf.ecore.impl.MinimalEObjectImpl$Container"
importerID="org.eclipse.emf.importer.ecore"
    complianceLevel="7.0" copyrightFields="false" operationReflection="true"
importOrganizing="true">
  <foreignModel>SimpleRDBMS.ecore</foreignModel>
  <genPackages prefix="PrimitiveTypes" disposableProviderFactory="true"
ecorePackage="SimpleRDBMS.ecore#/0">
    <genDataTypes ecoreDataType="SimpleRDBMS.ecore#/0/String"/>
  </genPackages>
  <genPackages prefix="SimpleRDBMS" disposableProviderFactory="true"
ecorePackage="SimpleRDBMS.ecore#/1">
    <genClasses ecoreClass="SimpleRDBMS.ecore#/1/Table">
      <genFeatures createChild="false" ecoreFeature="ecore:EAttribute
SimpleRDBMS.ecore#/1/Table/name"/>
      <genFeatures property="None" children="true" createChild="true"
ecoreFeature="ecore:EReference SimpleRDBMS.ecore#/1/Table/fkeys"/>
      <genFeatures notify="false" createChild="false" propertySortChoices="true"
ecoreFeature="ecore:EReference SimpleRDBMS.ecore#/1/Table/pkey"/>
      <genFeatures property="None" children="true" createChild="true"
ecoreFeature="ecore:EReference SimpleRDBMS.ecore#/1/Table/cols"/>
      <genFeatures createChild="false" ecoreFeature="ecore:EAttribute
SimpleRDBMS.ecore#/1/Table/tipo"/>
    </genClasses>
    <genClasses ecoreClass="SimpleRDBMS.ecore#/1/Column">
      <genFeatures createChild="false" ecoreFeature="ecore:EAttribute
SimpleRDBMS.ecore#/1/Column/name"/>
      <genFeatures createChild="false" ecoreFeature="ecore:EAttribute
SimpleRDBMS.ecore#/1/Column/type"/>
    </genClasses>
    <genClasses ecoreClass="SimpleRDBMS.ecore#/1/FKey">
      <genFeatures notify="false" createChild="false" propertySortChoices="true"
ecoreFeature="ecore:EReference SimpleRDBMS.ecore#/1/FKey/references"/>
      <genFeatures notify="false" createChild="false" propertySortChoices="true"
ecoreFeature="ecore:EReference SimpleRDBMS.ecore#/1/FKey/cols"/>
    </genClasses>
  </genPackages>
</genmodel:GenModel>
```

*Tabla A-4. Código SimpleRDBMS.genmodel*

```
<?xml version="1.0" encoding="ASCII"?>
<xmi:XMI xmi:version="2.0"
    xmlns:xmi="http://www.omg.org/XMI"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:SimpleClass="sedici.Simple"
    xsi:schemaLocation="sedici.Simple ../Metamodelos/SimpleClass.ecore#/0">
  <SimpleClass:Class
      name="Item"
      is_persistent="true"
      tipo="Normal">
    <attrs name="titulo"
        is_primary="false"
        type="/4"/>
    <attrs name="ID"
        is_primary="true"
        type="/3"/>
    <attrs name="tipo"
        is_primary="false"
        type="/7"/>
  </SimpleClass:Class>
  <SimpleClass:Class
      name="metadatos"
      is_persistent="true"
      tipo="Meta">
    <attrs name="name"
        is_primary="false"
```

```
        type="/4"/>
    <attrs name="ID"
        is_primary="true"
        type="/3"/>
    <attrs name="valor"
        is_primary="false"
        type="/4"/>
    <attrs name="tipo"
        is_primary="false"
        type="/8"/>
    <attrs name="categoria"
        is_primary="false"
        type="/10"/>
  </SimpleClass:Class>
  <SimpleClass:Class
      name="autores"
      is_persistent="true"
      tipo="EA">
    <attrs name="ID"
        is_primary="true"
        type="/3"/>
    <attrs name="name"
        is_primary="false"
        type="/4"/>
    <attrs name="valor"
        is_primary="false"
        type="/4"/>
  </SimpleClass:Class>
  <SimpleClass:PrimitiveDataType
      name="Integer"/>
  <SimpleClass:PrimitiveDataType
      name="String"/>
  <SimpleClass:PrimitiveDataType
      name="Long"/>
  <SimpleClass:PrimitiveDataType
      name="Boolean"/>
  <SimpleClass:PrimitiveDataType
      name="TypesCI"/>
  <SimpleClass:PrimitiveDataType
      name="Vocabulary"/>
  <SimpleClass:PrimitiveDataType
      name="AE"/>
  <SimpleClass:PrimitiveDataType
      name="Classification"/>
  <SimpleClass:Association
      src="/2"
      dest="/1"
      name="entidades"/>
  <SimpleClass:Association
      src="/1"
      dest="/0"
      name="item"/>
  <SimpleClass:Class
      name="File"
      is_persistent="true"
      tipo="Normal">
    <attrs name="name"
        is_primary="false"
        type="/4"/>
    <attrs name="handle"
        is_primary="true"
        type="/3"/>
  </SimpleClass:Class>
  <SimpleClass:Association
      src="/13"
      dest="/0"
      name="archivos"/>
</xmi:XMI>
```

*Tabla A-5. Código ejemplo de norma.XMI*

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:SimpleRDBMS="sedici.SimpleRDBMS">
  <SimpleRDBMS:Table name="Item" pkey="/0/@cols.0" tipo="Normal">
    <cols name="ID" type="Integer"/>
    <cols name="titulo" type="String"/>
    <cols name="tipo" type="TypesCI"/>
  </SimpleRDBMS:Table>
  <SimpleRDBMS:Table name="metadatos" pkey="/1/@cols.0" tipo="Meta">
    <fkeys references="/0" cols="/1/@cols.1"/>
    <cols name="ID" type="Integer"/>
    <cols name="item_ID" type="Integer"/>
    <cols name="name" type="String"/>
    <cols name="valor" type="String"/>
    <cols name="tipo" type="Vocabulary"/>
    <cols name="categoria" type="Classification"/>
  </SimpleRDBMS:Table>
  <SimpleRDBMS:Table name="autores" pkey="/2/@cols.0" tipo="EA">
    <fkeys references="/1" cols="/2/@cols.1"/>
    <cols name="ID" type="Integer"/>
    <cols name="entidades_ID" type="Integer"/>
    <cols name="name" type="String"/>
    <cols name="valor" type="String"/>
  </SimpleRDBMS:Table>
  <SimpleRDBMS:Table name="File" pkey="/3/@cols.0" tipo="Normal">
    <fkeys references="/0" cols="/3/@cols.1"/>
    <cols name="handle" type="Integer"/>
    <cols name="archivos_ID" type="Integer"/>
    <cols name="name" type="String"/>
  </SimpleRDBMS:Table>
</xmi:XMI>
```

*Tabla A-6. Código del ejemplo de basededatos.XMI*

```
module ATL_Curso;
create OUT : SimpleRDBMS from IN : SimpleClass;

rule PersistentClass2Table{
        from
                c : SimpleClass!Class (
                          c.is_persistent and c.parent->oclIsUndefined()
                        )
        using {
                primary_attributes : Sequence(TupleType(name : String,
                                                        type : SimpleClass!Classifier,
                                                        isPrimary : Boolean)
                                    ) =
                                c.flattenedFeatures->select(f | f.isPrimary);
                persistent_features : Sequence(TupleType(
                                                        name : String,
                                                        class : SimpleClass!Class,
                                                        offcet : Integer,
                                                               nofAttrs : Integer
                                                )
                                     ) =
                c.flattenedFeatures->iterate(tuple; acc : Sequence(TupleType(name :
String,                                                 class : SimpleClass!Class,
                                                        offcet : Integer,
                                                               nofAttrs :
Integer))=Sequence{}|
                                if tuple.type->oclIsKindOf(SimpleClass!Class)
                                        then
                                                acc->append(
                                                        Tuple{
                                                        name=tuple.name,
                                                        class = tuple.type,
                                                        offcet=if acc->size()=0 then 1
```

```
                                                    else acc->last().offcet + acc-
>last().nofAttrs
                                                    endif,
nofAttrs=tuple.type.topParent.flattenedFeatures-
                                                >select(t | t.isPrimary)->size()
                                            }
                                        )
                                                else
                                                        acc
                                                endif
                        );
                foreign_key_attributes : Sequence(TupleType(name : String,
                                                            type : SimpleClass!
Classifier)
                                        ) =
                                    persistent_features->collect(tuple |
                                      tuple.class.topParent.flattenedFeatures->select(t |
                                      t.isPrimary)->collect(a |
                                    Tuple {
                                       name=tuple.name + '_' + a.name, type=a.type
                                                }
                                      )
                                   )->flatten();
                rest_of_attributes : Sequence(TupleType(name : String,
                                                type : SimpleClass!Classifier
                                                 )) =
                        c.flattenedFeatures->select(tuple |
                                        not tuple.isPrimary and
                                        not tuple.type->oclIsKindOf(SimpleClass!Class)
                                                 );
        }

        to
                t : SimpleRDBMS!Table (
                        name<-c.name,
                        tipo<-c.tipo,
                        cols<-primary_key_columns->union(foreign_key_columns)-
>union(rest),
                        pkey<-primary_key_columns,
                        fkeys<-foreign_keys
                        ),

                primary_key_columns : distinct SimpleRDBMS!Column foreach (primAttr in
                                                            primary_attributes)
                         (
                                  name<-primAttr.name,
                                  type<-primAttr.type.name
                          ),
                foreign_keys : distinct SimpleRDBMS!FKey foreach (persAttr in
                                                            persistent_features)
                        (
                                references<-persAttr.class.topParent,
                                cols<-persistent_features->iterate(tuple; acc :
                                       Sequence(Sequence(SimpleRDBMS!Column))=Sequence{} |
                                        acc->append(foreign_key_columns.subSequence(
                                              tuple.offcet,
                                              tuple.offcet + tuple.nofAttrs-1)
                                             )
                                          )
                                ),

                foreign_key_columns : distinct SimpleRDBMS!Column foreach (attr in
                                                            foreign_key_attributes)
                        (
                        name<-attr.name,
                        type<-attr.type.name
                        ),
```

```
                rest : distinct SimpleRDBMS!Column foreach (attr in rest_of_attributes)
                        (
                        name<-attr.name,
                        type<-attr.type.name
                        )
}

helper context SimpleClass!Class def :
        allAttributes : Sequence(SimpleClass!Attribute) =
        self.attrs->union(
                if not self.parent.oclIsUndefined() then
                        self.parent.allAttributes->select(attr |
                           not self.attrs->exists(at | at.name = attr.name)
                        )
                else
                        Sequence {}
                endif
        )->flatten();

helper context SimpleClass!Class def :
        allAssociations : Sequence(SimpleClass!Association) =
        let defAssoc : Sequence(SimpleClass!Association) =
                SimpleClass!Association.allInstances()->select(assoc |
                        assoc.src = self) in
        defAssoc->union(
                if not self.parent.oclIsUndefined() then
                   self.parent.allAssociations
                else
                   Sequence {}
         endif
        )->flatten();

helper context SimpleClass!Class def :
        attributesOfSubclasses : Sequence(SimpleClass!Attribute) =
        let attrsInSubclasses : Sequence(SimpleClass!Attribute) =
                SimpleClass!Class.allInstances()->select(c |
                        c.parent=self
                )->collect(directSubclass |
                        directSubclass.attributesOfSubclasses
                )->flatten() in
        attrsInSubclasses->union(
                self.attrs->select(attr |
                        not attrsInSubclasses->exists(a |
                                a.name = attr.name)
        ))->flatten();

helper context SimpleClass!Class def :
        associationsOfSubclasses : Sequence(SimpleClass!Association) =
        SimpleClass!Association.allInstances()->select(assoc |
                        assoc.src = self)->union(
                        SimpleClass!Class.allInstances()->select(c |
                                c.parent = self)->collect(subclass |
                        subclass.associationsOfSubclasses)->flatten()
        )->flatten();

helper context SimpleClass!Class def :
        topParent : SimpleClass!Class =
        if self.parent.oclIsUndefined() then
                self
        else
                self.parent.topParent
        endif;

helper context SimpleClass!Class def :
        flattenedFeatures : Sequence(TupleType(
                                        name : String,
                                        type : SimpleClass!Classifier,
                                        isPrimary : Boolean
                                )) =
    if self.topParent.is_persistent then
```

```
                self.topParent.attributesOfSubclasses->union(
                        self.topParent.associationsOfSubclasses)
        else
                self.allAttributes->union(self.allAssociations)
        endif->collect(f |
                let feature : TupleType(
                                        name : String,
                                        type : SimpleClass!Classifier,
                                       isPrimary : Boolean
                                    ) =
                if f.oclIsKindOf(SimpleClass!Attribute) then
                        Tuple{name = f.name, type = f.type, isPrimary = f.is_primary}
                else
                        Tuple{name = f.name, type = f.dest, isPrimary = false}
                endif in
        if feature.type.oclIsKindOf(SimpleClass!PrimitiveDataType) then
                feature
        else if not feature.type.topParent.is_persistent then
                feature.type.flattenedFeatures->collect (f | Tuple{name=feature.name+
'_' +
                                        f.name, type=f.type, isPrimary=f.isPrimary})
        else feature
        endif endif
        )->flatten();
```

*Tabla A-7. Código de transformación ATL*

## Código fuente – Fase 3

El código de esta sección corresponde a la transformación de modelo a texto (M2T) que se realizó con el lenguaje Acceleo, todo el código fuente se encuentra en un proyecto GitHub Tesis/Fase3 (GitHub, 2014b). La estructura principal de archivos del proyecto Acceleo es:

```
/Thesis/                          ==>  proyecto principal donde se encuentran los metamodelos
....../SimpleRDBMS.ecore          ==>  metamodelo base de entrada
....../SimpleRDBMS.ecorediag
....../SimpleRDBMS.genmodel       ==>  Tabla A-8
....../basededatos.XMI
/Thesis.edit/
/Thesis.editor/
/Thesis.tests/
/runtime-EclipseApplication/      ==>  proyecto donde se realiza la transformación M2T
....../generate.mtl               ==>  Tabla A-9
....../crear.SQL                  ==>  Tabla A-10
```

```
<?xml version="1.0" encoding="UTF-8"?>
<genmodel:GenModel          xmi:version="2.0"          xmlns:xmi="http://www.omg.org/XMI"
xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
                            xmlns:genmodel="http://www.eclipse.org/emf/2002/GenModel"
modelDirectory="/Thesis/src" modelPluginID="Thesis" modelName="RDBMS"
             rootExtendsClass="org.eclipse.emf.ecore.impl.MinimalEObjectImpl$Container"
importerID="org.eclipse.emf.importer.ecore"
           complianceLevel="7.0"   copyrightFields="false"   operationReflection="true"
importOrganizing="true">
  <foreignModel>RDBMS.ecore</foreignModel>
                <genPackages        prefix="PT"        disposableProviderFactory="true"
ecorePackage="RDBMS.ecore#/0">
    <genDataTypes ecoreDataType="RDBMS.ecore#/0/String"/>
  </genPackages>
              <genPackages       prefix="RDBMS"       disposableProviderFactory="true"
ecorePackage="RDBMS.ecore#/1">
    <genClasses ecoreClass="RDBMS.ecore#/1/Table">
                    <genFeatures   createChild="false"   ecoreFeature="ecore:EAttribute
RDBMS.ecore#/1/Table/name"/>
                  <genFeatures   property="None"   children="true"   createChild="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/Table/fkeys"/>
          <genFeatures  notify="false"  createChild="false"  propertySortChoices="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/Table/pkey"/>
                  <genFeatures   property="None"   children="true"   createChild="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/Table/cols"/>
                    <genFeatures   createChild="false"   ecoreFeature="ecore:EAttribute
RDBMS.ecore#/1/Table/tipo"/>
    </genClasses>
    <genClasses ecoreClass="RDBMS.ecore#/1/Column">
                    <genFeatures   createChild="false"   ecoreFeature="ecore:EAttribute
RDBMS.ecore#/1/Column/name"/>
                    <genFeatures   createChild="false"   ecoreFeature="ecore:EAttribute
RDBMS.ecore#/1/Column/type"/>
    </genClasses>
    <genClasses ecoreClass="RDBMS.ecore#/1/FKey">
          <genFeatures  notify="false"  createChild="false"  propertySortChoices="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/FKey/references"/>
          <genFeatures  notify="false"  createChild="false"  propertySortChoices="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/FKey/cols"/>
    </genClasses>
    <genClasses ecoreClass="RDBMS.ecore#/1/Schema">
                    <genFeatures   createChild="false"   ecoreFeature="ecore:EAttribute
```

```
RDBMS.ecore#/1/Schema/name"/>
                <genFeatures   property="None"   children="true"   createChild="true"
ecoreFeature="ecore:EReference RDBMS.ecore#/1/Schema/tables"/>
    </genClasses>
  </genPackages>
</genmodel:GenModel>
```

*Tabla A-8. Código SimpleRDBMS.genmodel*

```
[comment encoding = UTF-8 /]
[module generate('http://ISO.RDBMS/2.0')]

[comment [import  /]

[template public generateElement(aSc : Schema)]
[comment @main/]

[file (aSc.name.concat('.SQL'), false, 'UTF-8')]

[for (aT : Table | aSc.tables)]

CREATE TABLE [aT.name/] (

[for (aC : Column | aT.cols)]
       COLUM [aC.name/] [aC.type/],
[/for]


);
[/for]

[/file]
[/template]
```

*Tabla A-9. Código ejemplo de la transformación M2T*

```
CREATE TABLE A (

       COLUM c1 String,
       COLUM c2 String,

);

CREATE TABLE B (

       COLUM name String,
);

CREATE TABLE C (

       COLUM nombre String,
       COLUM valor String,

);
```

*Tabla A-10. Código ejemplo de salida de M2T – crear.SQL*

## Código fuente – Fase 4

El código descrito en esta sección corresponde a la fase de transformación de las múltiples entradas de archivos a un script de incorporación de recursos. Todo el código fuente de esta fase se encuentra en el proyecto GitHub Tesis/Fase4 (GitHub, 2014b). La estructura principal de archivos del proyecto Java es:

```
/Inserts/
....../crear.SQL
....../autores.in
....../comunidades.in
....../colecciones.in
....../instituciones.in
....../revistas.in
....../num_revistas.in
/src/t2t/
....../T2T_insert.java        ==>  Tabla A-11
```

```
package t2t;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.w3c.dom.Document;
import org.w3c.dom.NodeList;
import org.w3c.dom.Node;
import org.w3c.dom.Element;
import java.io.File;

public class T2T_insert {

 /**
  * @param args
  */
 public static void main(String[] args) {
       // TODO Auto-generated method stub
     try {
       File fXmlFile = new File("schema_db.xmi");
       DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
       DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
       Document doc = dBuilder.parse(fXmlFile);
       doc.getDocumentElement().normalize();
       System.out.println("Root element :" + doc.getDocumentElement().getNodeName());
       NodeList nList = doc.getElementsByTagName("SimpleRDBMS:Table");
       System.out.println(nList.getLength());
       for (int temp = 0; temp < nList.getLength(); temp++) {
              Node nNode = nList.item(temp);
              System.out.println("\nCurrent Element :" + nNode.getNodeName());
              if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                     Element eElement = (Element) nNode;
                     for (int j = 0;
j<eElement.getElementsByTagName("cols").getLength(); j++)
                            {
                            System.out.println("\t"+


eElement.getElementsByTagName("cols").item(j).getAttributes().getNamedItem("name").getN
odeValue()
                                                 );
                            }

 System.out.println("\tFKEY:"+eElement.getElementsByTagName("fkeys").getLength());
                     for (int j = 0;
j<eElement.getElementsByTagName("fkeys").getLength(); j++)
```

```
                                {
                                System.out.println("\t \t"+


eElement.getElementsByTagName("fkeys").item(j).getAttributes().getNamedItem("references
").getNodeValue()+
                                                        " with "+


eElement.getElementsByTagName("fkeys").item(j).getAttributes().getNamedItem("cols").get
NodeValue()
                                                        );
                                }
                            }
                        }
                    }
                catch (Exception e) {
                  e.printStackTrace();
                    }
                }
         }
```

*Tabla A-11. Código de transformación T2T*

## Código fuente – Fase 5

El código fuente correspondiente al proyecto en WebRatio se encuentra en un proyecto GitHub con el título Tesis/Fase5 (GitHub, 2014b) y la estructura principal de archivos del proyecto es:

```
/RR/
....../Model.wr                              ==> Tabla A-12
....../sedici.xsd                            ==> Tabla 5-6
/Config/
/DBScripts/
/Model/
...../WebModel/Properties.wr                 ==> Tabla A-13
...../WebModel/sv1/area1/
.......................page1.wr              ==> Tabla A-14
.......................Properties.wr
...../WebModel/sv1/area2/
.......................page1.wr
.......................Properties.wr
...../WebModel/sv1/area3/
.......................page1.wr
.......................Properties.wr
...../WebModel/sv1/area4/
.......................page1.wr
.......................Properties.wr
.....
...../DataModel/Properties.wr                ==> Tabla A-15
/WebContent/
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?webml version="7.2.0.1"?>

<WebProject xmlns:gr="http://www.webratio.com/2006/WebML/Graph" outputPath="$
{webapps_loc}/${project_name}" gr:showUnitContent="true" gr:showTooltip="true"
httpPort="8080" httpsPort="8443" enableProjectDependencies="false"
jobGroupName="WEBRATIO" wrTempNewProject="true">
  <ServiceDataProviders>
    <XsdProvider id="xsdp1" name="XSD Provider 1">
      <XsdResource id="xsdp1#xsdr1" name="sedici" file="sedici.xsd"/>
    </XsdProvider>
  </ServiceDataProviders>
</WebProject>
```

*Tabla A-12. Código RR/Model.wr*

```
<WebModel xmlns:layout="http://www.webratio.com/2006/WebML/Layout" defaultLocale="lcl1"
layout:style="WebRatio" layout:inheritParameters="true"
control="Q291bnRyeUlTT0N0eFBhcmFtR3JvdXBDdHhQYXJhbUxhbmd1YWdlSVNPQ3R4UGFyYW1Vc2VyQ3R4UG
FyYW0=" homeSiteView="sv1">
  <Locale id="lcl1" country="US" language="en">
    <PatternConfiguration type="boolean" pattern="yes|no"/>
    <PatternConfiguration type="decimal" useNumberPattern="false" maxDecimal="3"
minDecimal="0" minInteger="1" useThousandSeparator="true"/>
    <PatternConfiguration type="date" pattern="M/d/yy"/>
    <PatternConfiguration type="float" useNumberPattern="false" maxDecimal="3"
minDecimal="0" minInteger="1" useThousandSeparator="true"/>
    <PatternConfiguration type="integer" useNumberPattern="false" minInteger="1"
useThousandSeparator="true"/>
    <PatternConfiguration type="time" pattern="h:mm:ss a"/>
```

```
    <PatternConfiguration type="timestamp" pattern="M/d/yy h:mm:ss a"/>
  </Locale>
  <ContextParameter id="UserCtxParam" name="UserCtxParam" entity="User" type="entity"/>
  <ContextParameter id="GroupCtxParam" name="GroupCtxParam" entity="Group"
type="entity"/>
  <ContextParameter id="LanguageISOCtxParam" name="LanguageISOCtxParam" type="string"/>
  <ContextParameter id="CountryISOCtxParam" name="CountryISOCtxParam" type="string"/>
</WebModel>
```

*Tabla A-13. Código WebModel/Properties.wr*

```
<Page xmlns:gr="http://www.webratio.com/2006/WebML/Graph"
xmlns:layout="http://www.webratio.com/2006/WebML/Layout" gr:x="0" gr:y="0"
id="sv1#area2#page2" name="All..." linkOrder="sv1#area2#page2#ln2">
  <ContentUnits>
    <IndexUnit gr:x="0" gr:y="160" id="sv1#area2#page2#inu1" name="details..."
entity="ent5" displayAttributes="ent5#att20 rel3#role5.ent7#att26
rel3#role5.ent7#att27">
      <Selector id="sv1#area2#page2#inu1#su2" defaultPolicy="fill"
booleanOperator="and">
        <AttributesCondition id="sv1#area2#page2#inu1#su2#acond2" name="AttCondition2"
predicate="eq" booleanOperator="or" implied="false" value="64|25|66|94|130|146|132|121|
89|85|117|118|119|120" attributes="rel3#role5.ent7#att25"/>
        <RelationshipRoleCondition id="sv1#area2#page2#inu1#su2#rcond1"
name="RoleCondition1" predicate="in" implied="false" role="rel2#role4"/>
      </Selector>
      <SortAttribute attribute="ent5#att19" order="ascending"/>
    </IndexUnit>
    <PowerIndexUnit gr:x="15" gr:y="0" id="sv1#area2#page2#pwu1" name="todos..."
sortable="true" checkable="false" useCountQuery="false" entity="ent5"
displayAttributes="ent5#att20 rel2#role3.ent4#att17 rel2#role3.ent4#att16"
blockFactor="10" blockWindow="4" maxResults="100" linkOrder="sv1#area2#page2#pwu1#ln4">
      <SortAttribute attribute="rel2#role3.ent4#att16" order="ascending"/>
      <SortAttribute attribute="ent5#att20" order="ascending"/>
      <Selector id="sv1#area2#page2#pwu1#su1" defaultPolicy="fill"
booleanOperator="and">
        <AttributesCondition id="sv1#area2#page2#pwu1#su1#acond1" name="AttCondition1"
predicate="eq" booleanOperator="or" implied="false" value="64"
attributes="rel3#role5.ent7#att25"/>
      </Selector>
      <Link id="sv1#area2#page2#pwu1#ln4" name="Flow4" to="sv1#area2#page2#inu1"
type="normal" validate="true">
        <LinkParameter id="sv1#area2#page2#pwu1#ln4#par1"
name="MetadatosToItems.id_RoleCondition1 [id]"
target="sv1#area2#page2#inu1#su2#rcond1.ent4#att16"
source="data[].rel2#role3_ent4#att16"/>
      </Link>
    </PowerIndexUnit>
  </ContentUnits>
  <layout:Grid containerType="grid">
    <layout:Row>
      <layout:Cell colspan="12">
        <layout:Unit unitId="sv1#area2#page2#pwu1">
          <layout:Link link="sv1#area2#page2#pwu1#ln4"/>
          <layout:Link link="sv1#area2#page2#pwu1rel2#role3$ent4#att16"/>
          <layout:Link link="sv1#area2#page2#pwu1ent5#att20"/>
          <layout:Link link="sv1#area2#page2#pwu1First"/>
          <layout:Link link="sv1#area2#page2#pwu1Previous"/>
          <layout:Link link="sv1#area2#page2#pwu1Block"/>
          <layout:Link link="sv1#area2#page2#pwu1Next"/>
          <layout:Link link="sv1#area2#page2#pwu1Last"/>
          <layout:Attribute attribute="ent5#att20"/>
          <layout:Attribute attribute="rel2#role3.ent4#att17"/>
          <layout:Attribute attribute="rel2#role3.ent4#att16"/>
        </layout:Unit>
      </layout:Cell>
      <layout:Cell/>
      <layout:Cell/>
```

```
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
    </layout:Row>
    <layout:Row>
      <layout:Cell colspan="12">
        <layout:Unit unitId="sv1#area2#page2#inu1">
          <layout:Attribute attribute="ent5#att20"/>
          <layout:Attribute attribute="rel3#role5.ent7#att26"/>
          <layout:Attribute attribute="rel3#role5.ent7#att27"/>
        </layout:Unit>
      </layout:Cell>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
      <layout:Cell/>
    </layout:Row>
  </layout:Grid>
  <Link id="sv1#area2#page2#ln2" name="Inicio" to="sv1#area1#page1"
automaticCoupling="true" type="normal" validate="true" gr:bendpoints="-202,-
48,183,72"/>
</Page>
```

*Tabla A-14. Código WebModel/sv1/area1/page1.wr*

```
<DataModel xmlns:db="http://www.webratio.com/2006/WebML/Database"
xmlns:gr="http://www.webratio.com/2006/WebML/Graph"
xmlns:wrxsd="http://www.webratio.com/2012/WebML/WRXSD">
  <Entity id="User" name="User" duration="persistent" gr:x="30" gr:y="455"
gr:hideAttributes="false" attributeOrder="userOID userName password email"
db:database="db1" db:table="user">
    <Attribute name="oid" id="userOID" type="integer" key="true" db:column="oid"/>
    <Attribute name="userName" id="userName" type="string" db:column="username"/>
    <Attribute name="password" id="password" type="password" db:column="password"/>
    <Attribute name="email" id="email" type="string" db:column="email"/>
  </Entity>
  <Entity id="Group" name="Group" duration="persistent" gr:x="290" gr:y="475"
gr:hideAttributes="false" attributeOrder="groupOID groupName" db:database="db1"
db:table="group">
    <Attribute name="oid" id="groupOID" type="integer" key="true" db:column="oid"/>
    <Attribute name="groupName" id="groupName" type="string" db:column="groupname"/>
  </Entity>
  <Entity id="Module" name="Module" duration="persistent" gr:x="525" gr:y="465"
gr:hideAttributes="false" attributeOrder="moduleOID moduleID moduleName"
db:database="db1" db:table="module">
    <Attribute name="oid" id="moduleOID" type="integer" key="true" db:column="oid"/>
    <Attribute name="moduleID" id="moduleID" type="string" db:column="moduleid"/>
    <Attribute name="moduleName" id="moduleName" type="string" db:column="modulename"/>
  </Entity>
  <Relationship id="User2Group_Group2User" name="User_Group" sourceEntity="User"
targetEntity="Group" db:database="db1" db:table="user_group">
    <RelationshipRole1 id="User2Group" name="groups" maxCard="N">
      <db:JoinColumn attribute="userOID" name="user_oid"/>
    </RelationshipRole1>
    <RelationshipRole2 id="Group2User" name="users" maxCard="N">
```

```
      <db:JoinColumn attribute="groupOID" name="group_oid"/>
    </RelationshipRole2>
  </Relationship>
  <Relationship id="User2DefaultGroup_DefaultGroup2User" name="User_DefaultGroup"
sourceEntity="User" targetEntity="Group" gr:bendpoints="113,42,-112,44"
db:database="db1" db:table="user">
    <RelationshipRole1 id="User2DefaultGroup" name="defaultGroup" maxCard="1">
      <db:JoinColumn attribute="userOID" name="oid"/>
    </RelationshipRole1>
    <RelationshipRole2 id="DefaultGroup2User" name="defaultUsers" maxCard="N">
      <db:JoinColumn attribute="groupOID" name="group_oid"/>
    </RelationshipRole2>
  </Relationship>
  <Relationship id="Group2DefaultModule_DefaultModule2Group" name="Group_DefaultModule"
sourceEntity="Group" targetEntity="Module" db:database="db1" db:table="group">
    <RelationshipRole1 id="Group2DefaultModule" name="defaultModule" maxCard="1">
      <db:JoinColumn attribute="groupOID" name="oid"/>
    </RelationshipRole1>
    <RelationshipRole2 id="DefaultModule2Group" name="defaultGroups" maxCard="N">
      <db:JoinColumn attribute="moduleOID" name="module_oid"/>
    </RelationshipRole2>
  </Relationship>
  <Relationship id="Group2Module_Module2Group" name="Group_Module" sourceEntity="Group"
targetEntity="Module" gr:bendpoints="110,41,-115,41" db:database="db1"
db:table="group_module">
    <RelationshipRole1 id="Group2Module" name="modules" maxCard="N">
      <db:JoinColumn attribute="groupOID" name="group_oid"/>
    </RelationshipRole1>
    <RelationshipRole2 id="Module2Group" name="groups" maxCard="N">
      <db:JoinColumn attribute="moduleOID" name="module_oid"/>
    </RelationshipRole2>
  </Relationship>
  <db:Database id="db1" name="RR_SEDICI" type="PostgreSQL 9"
url="jdbc:postgresql://localhost:5432/Propuesta" username="postgres"
password="MzIxNjg5MDA=" cryptedPassword="true" connectionCount="5" schema="public"/>
  <Entity id="ent1" name="Autor" db:database="db1" duration="persistent"
db:table="Autor" gr:x="260" gr:y="0">
    <Attribute id="ent1#att1" name="idA" db:column="ID_A" type="integer" key="true"/>
    <Attribute id="ent1#att2" name="nombre" db:column="nombre" type="string"
key="false"/>
    <Attribute id="ent1#att3" name="apellido" db:column="apellido" type="string"
key="false"/>
    <Attribute id="ent1#att4" name="dni" db:column="dni" type="integer" key="false"/>
    <Attribute id="ent1#att5" name="email" db:column="email" type="string"
key="false"/>
    <Attribute id="ent1#att6" name="dependencia" db:column="dependencia" type="string"
key="false"/>
    <Attribute id="ent1#att7" name="unidad" db:column="unidad" type="string"
key="false"/>
    <Attribute id="ent1#att8" name="paisId" db:column="pais_ID" type="integer"
key="false"/>
  </Entity>
  <Entity id="ent2" name="Esquema" db:database="db1" duration="persistent"
db:table="Esquema" gr:x="505" gr:y="345">
    <Attribute id="ent2#att9" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent2#att10" name="nombre" db:column="nombre" type="string"
key="false"/>
    <Attribute id="ent2#att11" name="namespace" db:column="namespace" type="string"
key="false"/>
  </Entity>
  <Entity id="ent3" name="Institucion" db:database="db1" duration="persistent"
db:table="Institucion" gr:x="540" gr:y="40">
    <Attribute id="ent3#att12" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent3#att13" name="nombre" db:column="nombre" type="string"
key="false"/>
    <Attribute id="ent3#att14" name="abreviatura" db:column="abreviatura" type="string"
key="false"/>
    <Attribute id="ent3#att15" name="paisId" db:column="pais_ID" type="integer"
key="false"/>
  </Entity>
```

```
  <Entity id="ent4" name="Items" db:database="db1" duration="persistent"
db:table="Items" gr:x="45" gr:y="205">
    <Attribute id="ent4#att16" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent4#att17" name="titulo" db:column="titulo" type="string"
key="false"/>
    <Attribute id="ent4#att18" name="idTipo" db:column="ID_tipo" type="string"
key="false"/>
  </Entity>
  <Entity id="ent5" name="Metadatos" db:database="db1" duration="persistent"
db:table="Metadatos" gr:x="280" gr:y="215">
    <Attribute id="ent5#att19" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent5#att20" name="valor" db:column="valor" type="string"
key="false"/>
  </Entity>
  <Entity id="ent6" name="Numrevista" db:database="db1" duration="persistent"
db:table="Numrevista" gr:x="480" gr:y="195">
    <Attribute id="ent6#att21" name="volumen" db:column="volumen" type="string"
key="false"/>
    <Attribute id="ent6#att22" name="numero" db:column="numero" type="string"
key="false"/>
    <Attribute id="ent6#att23" name="revistaId" db:column="revista_ID" type="integer"
key="false"/>
    <Attribute id="ent6#att24" name="editores" db:column="editores" type="string"
key="false"/>
    <Generalization superEntity="ent8">
      <db:JoinColumn attribute="ent8#att29" name="ID"/>
    </Generalization>
  </Entity>
  <Entity id="ent7" name="RegistroMetadato" db:database="db1" duration="persistent"
db:table="Registro_metadato" gr:x="260" gr:y="335">
    <Attribute id="ent7#att25" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent7#att26" name="element" db:column="element" type="string"
key="false"/>
    <Attribute id="ent7#att27" name="qualifier" db:column="qualifier" type="string"
key="false"/>
    <Attribute id="ent7#att28" name="scopeNote" db:column="scope_note" type="string"
key="false"/>
  </Entity>
  <Entity id="ent8" name="Revista" db:database="db1" duration="persistent"
db:table="Revista" gr:x="740" gr:y="215">
    <Attribute id="ent8#att29" name="id" db:column="ID" type="integer" key="true"/>
    <Attribute id="ent8#att30" name="titulo" db:column="titulo" type="string"
key="false"/>
  </Entity>
  <Relationship id="rel1" name="RegistrometadatoEsquema" db:database="db1"
db:table="Registro_metadato" sourceEntity="ent7" targetEntity="ent2">
    <RelationshipRole1 id="rel1#role1" name="RegistrometadatoToEsquema" maxCard="1">
      <db:JoinColumn attribute="ent7#att25" name="ID"/>
    </RelationshipRole1>
    <RelationshipRole2 id="rel1#role2" name="EsquemaToRegistrometadato" maxCard="N">
      <db:JoinColumn attribute="ent2#att9" name="esquema_ID"/>
    </RelationshipRole2>
  </Relationship>
  <Relationship id="rel2" name="MetadatosItems" db:database="db1" db:table="Metadatos"
sourceEntity="ent5" targetEntity="ent4">
    <RelationshipRole1 id="rel2#role3" name="MetadatosToItems" maxCard="1">
      <db:JoinColumn attribute="ent5#att19" name="ID"/>
    </RelationshipRole1>
    <RelationshipRole2 id="rel2#role4" name="ItemsToMetadatos" maxCard="N">
      <db:JoinColumn attribute="ent4#att16" name="item_ID"/>
    </RelationshipRole2>
  </Relationship>
  <Relationship id="rel3" name="MetadatosRegistrometadato" db:database="db1"
db:table="Metadatos" sourceEntity="ent5" targetEntity="ent7">
    <RelationshipRole1 id="rel3#role5" name="MetadatosToRegistrometadato" maxCard="1">
      <db:JoinColumn attribute="ent5#att19" name="ID"/>
    </RelationshipRole1>
    <RelationshipRole2 id="rel3#role6" name="RegistrometadatoToMetadatos" maxCard="N">
      <db:JoinColumn attribute="ent7#att25" name="registrometadato_ID"/>
    </RelationshipRole2>
```

```
  </Relationship>
  <Relationship id="rel4" name="AutorInstitucion" db:database="db1" db:table="Autor"
sourceEntity="ent1" targetEntity="ent3">
    <RelationshipRole1 id="rel4#role7" name="AutorToInstitucion" maxCard="1">
      <db:JoinColumn attribute="ent1#att1" name="ID_A"/>
    </RelationshipRole1>
    <RelationshipRole2 id="rel4#role8" name="InstitucionToAutor" maxCard="N">
      <db:JoinColumn attribute="ent3#att12" name="institucion_ID"/>
    </RelationshipRole2>
  </Relationship>
  <Package id="pkg1" name="xSDProvider1" gr:x="30" gr:y="575">
    <Package id="pkg1#pkg2" name="sedici" gr:x="0" gr:y="0">
      <Entity id="pkg1#pkg2#ent9" name="item" duration="volatile"
volatileStorage="database" wrxsd:provider="xsdp1" wrxsd:type="item[Element]" gr:x="0"
gr:y="0">
        <Attribute id="pkg1#pkg2#ent9#att31" name="oid" type="integer" key="true"/>
        <Attribute id="pkg1#pkg2#ent9#att32" name="title" type="string" key="false"
wrxsd:element="title"/>
        <Attribute id="pkg1#pkg2#ent9#att33" name="ID" type="integer" key="false"
wrxsd:element="id"/>
        <Attribute id="pkg1#pkg2#ent9#att34" name="tipo" type="string" key="false"
wrxsd:element="tipo"/>
      </Entity>
      <Entity id="pkg1#pkg2#ent10" name="item_metadatos" duration="volatile"
volatileStorage="database" wrxsd:provider="xsdp1" wrxsd:type="item/metadatos[Element]"
gr:x="210" gr:y="13">
        <Attribute id="pkg1#pkg2#ent10#att35" name="oid" type="integer" key="true"/>
        <Attribute id="pkg1#pkg2#ent10#att36" name="ID" type="integer" key="false"
wrxsd:element="id"/>
        <Attribute id="pkg1#pkg2#ent10#att37" name="valor" type="string" key="false"
wrxsd:element="valor"/>
      </Entity>
    </Package>
  </Package>
  <Relationship id="rel5" name="metadatos" sourceEntity="pkg1#pkg2#ent9"
targetEntity="pkg1#pkg2#ent10" wrxsd:element="metadatos">
    <RelationshipRole1 id="rel5#role9" name="metadatos" maxCard="1"/>
    <RelationshipRole2 id="rel5#role10" name="metadatosInverse" maxCard="1"/>
  </Relationship>
</DataModel>
```

*Tabla A-15. Código DataModel/Properties.wr*

# Apéndice B

Estadios de la investigación

En la Tabla B-1 se muestra los diferentes estadios de investigación para alcanzar el objetivo general del estudio realizado (Hurtado, 2008), *"Proponer un marco de referencia que permita definir un modelo general y flexible para obtener la representación de recursos del Repositorio Institucional de forma independiente de las tecnologías usadas"*:

| Niveles de conocimiento (estadio) | Acción / Logro | ¿Qué se desea? | Objetivos específicos |
|---|---|---|---|
| 1. Descriptivo | Describir el problema a resolver, es decir, el evento a modificar | ¿Qué ocurre?<br>¿Qué se quiere? | **Describir** la representación de los recursos dentro de un repositorio institucional que permita identificar las funciones básicas del repositorio. |
| 2. Analítico | Analizar las propuestas alternas (fortalezas y debilidades), donde la nueva propuesta las supere | ¿Qué se está haciendo?<br>¿Qué ventajas y debilidades tiene lo que hay? | **Analizar** y **comparar** las propuestas que se han desarrollado en cuanto a la representación de los recursos dentro de un repositorio institucional. |
| 3. Comparativo | Comparar las propuestas alternas | ¿Cuáles son los mejores aspectos de las propuestas alternas? | |
| 4. Explicativo | Identificar las causas o procesos generadores de la situación a resolver (explicar) | ¿Por qué ocurre la situación que se desea cambiar? | **Identificar** los factores que inciden en cada uno de los problemas de la representación de recursos dentro de un Repositorio Institucional.<br>**Caracterizar** los esquemas de metadatos, tipologías de los recursos y entidades abstractas existentes que afectan la representación de los recursos dentro de un repositorio institucional. |
| 5. Predictivo | Predecir o anticipar las tendencias futuras a fin que la propuesta no se desactualice en el tiempo | ¿Hacia dónde va la situación a modificar? | **Determinar** los posibles escenarios relacionados con las diferentes reglas de catalogación que pueden afectar la representación de los recursos en un repositorio institucional. |
| 6. Proyectivo | Diseñar la propuesta con base en la información obtenida en los pasos anteriores. | ¿Qué se va a hacer? | **Diseñar** un marco de referencia que soporte un modelo flexible de datos para la representación de los recursos en SEDICI.<br>**Implementar** un prototipo que permita mostrar una solución al problema planteado. |

*Tabla B-1. Niveles de conocimiento de la investigación*

# Apéndice C

Índice de acrónimos

- AACR2: segunda edición de las reglas de catalogación Anglo-American-Cataloguing
- ADF: análisis documental formal.
- ADC: análisis documental de contenido.
- AIP: paquete de información del archivo según la norma ISO 14721.
- BD: Bibliotecas Digitales.
- Celsius DL: desarrollo de software propio de un RI en SEDICI.
- DC: Dublin Core.
- DCMI: Dublin Core Metadata Initiative.
- DIP: paquete de información de diseminación según la norma ISO 14721.
- DSM: Domain Specific Modeling.
- DSL: Domain Specific Language.
- EA: Entidad Abstracta.
- ECDL: European Conference on Research and Advanced Technology for Digital Libraries
- EMF: Eclipse Modeling Framework.
- EMP: Eclipse Modeling Project.
- FRBR: Functional Requirements for Bibliographic Records.
- FRAD: Functional Requirements for Authority Data.
- FRSAD: Functional Requirements for Subject Authority Data.
- GMP: Graphical Modeling Project.
- GMF: Graphical Modeling Framework.
- ICADL: International Conference on Asian Digital Libraries
- IM: Implementation Model.
- IP: paquete de información según la norma ISO 14721.
- ISBD: International Standard Bibliographic Description.

- JCDL: Joint Conference on Digital Libraries.
- MADS: Metadata Authority Description Schema.
- MARC 21: MAchine-Readable Cataloging - Century 21st.
- MBE: Model-Based Engineering.
- MDA: Model-Driven Architecture.
- MDD: Model-Driven Development o Desarrollo de Software Dirigido por Modelos.
- MDE: Model-Driven Engineering.
- MDSE: Model Driven Software Engineering.
- MDPE: Model Driven Product Engineering.
- MDWE: Model Driven Web Engineering.
- MD*: Model Driven star.
- METS: Metadata Encoding and Transmission Standard.
- MODS: Metadata Object Description Schema.
- MOF: Meta-Object Facility.
- M2M: transformación de modelo a modelo.
- M2T: transformación de modelo a texto.
- LIS: Library & Information Science.
- OAI-PMH: Open Archives Initiative-Protocol Metadata Harvesting
- OAIS: Open Archival Information System según la norma ISO 14721.
- OMG: Object Management Group.
- OpenDOAR: Directory of Open Access Repositories.
- PIM: Platform Independent Model.
- PSM: Platform Specific Model.
- PREMIS: Preservation Metadata Implementation Strategies.
- RDA: Resource Description and Access.
- RI: Repositorio Institucional.
- RR: Representación de Recursos.
- ROAR: Registry of Open Access Repositories.

- SEDICI:  Servicio de Difusión de la Creación Intelectual.
- SIP: paquete de información  según la norma ISO 14721.
- SWORD: Simple Web-service Offering Repository Deposit.
- T2T: transformación de texto a texto.
- TIC: Tecnologías de Información y Comunicación
- TCDL: Technical Committee on Digital Libraries
- TPDL: Theory and Practice of Digital Libraries
- UML: Unified Modeling Language.
- XMI: XML Metadata Interchange.
- XML: Extensible Markup Language.
- XSD: XML Schema Definition.