An MDE-Based Graphical Tool for the Validation of MySQL Replication Models

Efraín Bautista, Nora La Serna
Unidad de Posgrado, Facultad de Ingeniería de Sistemas e Informática
Universidad Nacional Mayor de San Marcos
Lima, Perú
ebautistau@unmsm.edu.pe, nlasernap@unmsm.edu.pe

***Abstract*—At modeling level, diagramming tools such as Microsoft Visio are used to design MySQL replication models. However, this type of tools do not allow validating if the MySQL replication model is free of errors, showing errors if exists. Thus, we can have erroneous documentation of the MySQL replication models. Due to the lack of this feature, this is done manually, which becomes a tedious task, time consuming and error prone. This paper proposes a MDE-based graphical modeling tool under the Eclipse platform for the automatic validation of MySQL replication models. In addition, once a model has been validated, the tool is capable of generating the mysqlreplicate commands of configuration. The results of the experiments for the errors correction of MySQL replication models with 25 servers demonstrate that by using the proposed tool the time is reduced in more than 87% compared with the tool Microsoft Visio 2013.**

Keywords—MySQL Replication, MDE, DSL, DSM.

I. INTRODUCTION

La replicación de base de datos es el proceso de mantener múltiples copias de un servidor principal en diferentes ubicaciones físicas llamadas réplicas [1]. MySQL es considerado como uno de los sistemas de gestión de base de datos relacional de software libre (open source) más populares, exitosos y usados del mundo [2], [3], el cual soporta nativamente la replicación de base de datos, que es una técnica ampliamente usada por los administradores de base de datos (Database Administrators, DBAs) por ser la estrategia más común y económica para mejorar el rendimiento, escalabilidad y disponibilidad en aplicaciones basadas en MySQL [4], [5].

A nivel de modelado, diversas herramientas de diagramación, tales como Microsoft Visio [6], Dia [7], entre otras, son usadas para diseñar modelos de replicación MySQL. Sin embargo, este tipo de herramientas no permiten validar automáticamente si un modelo de replicación MySQL está libre de errores, lo que puede resultar tener documentación errónea de los modelos de replicación.

La falta de la funcionalidad antes mencionada conlleva a realizarla de forma manual, la cual se torna en una tarea tediosa, propensa a errores y que consume tiempo, más aún si el número de servidores MySQL del modelo es alto, con 15, 20, 25 o más servidores.

Por otro lado, la ingeniería dirigida por modelos (Model Driven Engineering, MDE) es un enfoque de la ingeniería de software, que se centra en la explotación de modelos como la piedra angular del proceso de desarrollo de software [8]. MDE propone especificar sistemas software mediante modelos a diferentes niveles de abstracción, de forma que los modelos de más alto nivel se transformen en otros de menor nivel hasta alcanzar tanto como sea posible un modelo ejecutable mediante la generación automática de código [9], [10]. MDE está basado en principios que involucra los conceptos de modelo, metamodelo, y transformaciones de modelos para la generación automática de código durante el proceso de desarrollo de software [8], [9]. En este sentido, los modelos y las transformaciones entre dichos modelos juegan un papel central en el proceso de desarrollo de software, convirtiéndose en la base de este paradigma [10].

En MDE, los lenguajes de dominio específico (Domain Specific Languages, DSLs) desempeñan un papel cada vez más importante [11], principalmente por su simplicidad (comparado a lenguajes de propósito general como UML), y enfoque en el dominio del problema [12]. Investigadores han mostrado mayor interés en DSLs gráficos, llamados también lenguajes de modelado de dominio específico (Domain Specific Modeling, DSM), porque son más expresivos que los DSLs textuales [13]. Los modelos diseñados en un DSL pueden ser validados y posteriormente usados en un proceso de transformación modelo a modelo y/o modelo a texto, la cual producen automáticamente otros modelos o artefactos basados en texto como código fuente o documentación [14].

En este trabajo se presenta el desarrollo de una herramienta DSL basada en MDE denominada MySQL Replication Modeling, que permita validar si un modelo de replicación MySQL está libre de errores, y adicionalmente generar a partir de un modelo de replicación válido los comandos mysqlreplicate de configuración.

El resto del artículo está organizado de la siguiente manera. En la Sección II se describe el marco teórico. En la sección III se presenta el estado del arte. En la sección IV se detalla el proceso de desarrollo de la herramienta. La sección V presenta un ejemplo ilustrativo de la herramienta. En la sección VI se presenta los experimentos y resultados. Finalmente en la sección VII se presenta las conclusiones y trabajos futuros.

978-1-4673-9143-6/15/$31.00

II. MARCO TEÓRICO

A. Replicación de Base de Datos

La replicación de base de datos es el proceso de mantener múltiples copias de un servidor principal en diferentes ubicaciones físicas llamadas réplicas [1], esta técnica es ampliamente usada principalmente para mejorar el rendimiento, escalabilidad y disponibilidad de los servidores de base de datos [15], [16], [17]. La replicación de base de datos mejora el rendimiento al reducir los tiempos de respuesta, esto se logra realizando un balanceo de carga de las transacciones entre todas las réplicas. También incrementa la disponibilidad mejorando la tolerancia a fallos, si el servidor principal falla debido a un error o por propósitos de mantenimiento, la información puede seguir siendo accesible en otras réplicas [1], [17], [18] [19].

B. Replicación en MySQL

MySQL soporta la replicación desde su versión 3.23.15 [20], que se liberó el 8 de Mayo de 2000 [21]. En la replicación MySQL a un servidor principal se le conoce como maestro, y a una réplica se le denomina esclavo [22], por tanto, la replicación en MySQL permite duplicar los datos desde un servidor maestro hacia uno o más servidores esclavos. MySQL soporta nativamente la replicación asíncrona como una característica estándar del motor de base de datos, dependiendo de la configuración, se puede replicar todas las bases de datos, algunas bases de datos, o incluso algunas tablas de una base de datos [17]. El servidor maestro registra en su log binario todos los cambios en los datos y objetos de la base de datos, luego estos datos son enviados y aplicados en los servidores esclavos. Desde la versión 5.6.5 de MySQL, la replicación está basada en identificadores globales de transacción (Global Transaction Identifiers, GTIDs), logrando mejorar la consistencia entre un servidor maestro y sus esclavos [22]. Mayores detalles de la replicación en MySQL se puede revisar en [4].

C. Modelado de la Replicación MySQL

Es posible modelar y configurar la replicación MySQL para las topologías mostradas en la Fig.1.

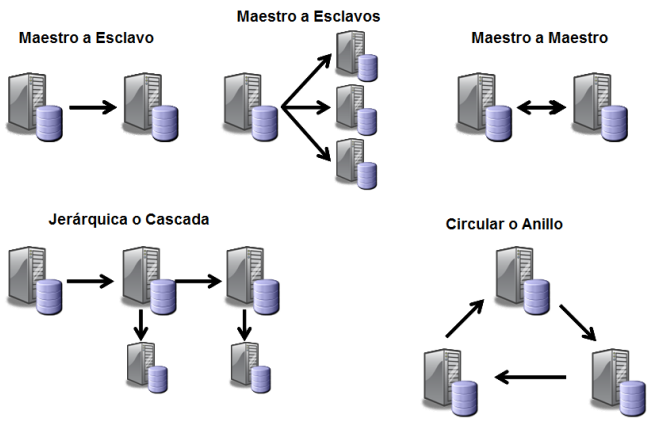


Fig. 1. Topologías de Replicación en MySQL [4]

A continuación se realiza una descripción breve de cada una de las topologías de replicación en MySQL [4].

1) Maestro a Esclavo: Esta topología es la más popular, fácil de configurar y administrar, en esta topología se tiene dos servidores, uno que actúa como servidor maestro y otro como servidor esclavo. Todas las transacciones de escritura son realizadas en el servidor maestro y las operaciones de lectura pueden ser realizadas tanto en el servidor maestro como en el servidor esclavo.

2) Maestro a Esclavos: En esta topología un servidor maestro tiene configurado varios servidores esclavos. Esto permite un mayor grado de escalabilidad pero tiene un costo mayor en administración. Los servidores esclavos presentan muy poca sobrecarga al servidor maestro, típicamente 1% de sobrecarga por cada servidor esclavo.

3) Replicación Jerárquica o en Cascada: Esta topología es una extensión de las topologías anteriores. En este caso, un servidor esclavo es asignado a otro servidor esclavo que actúa tanto como servidor maestro y esclavo puesto que está asignado a un servidor maestro principal. En esta configuración, todas las escrituras son realizadas en el servidor maestro principal.

4) Maestro a Maestro: En esta configuración, ambos servidores actúan como servidor maestro y esclavo. Se tiene la ventaja de poder realizar escrituras en cualquier servidor, pero esto aumenta el grado de complejidad en la instalación, configuración y administración. La aplicación debe asegurarse de no actualizar una fila mientras exista un cambio en la misma fila del otro servidor que no haya sido replicado aún. Al usar esta topología, las columnas autoincrement deben ser usadas con los parámetros auto_increment_offset y auto_increment_increment para asegurarse de que no existan valores duplicados.

5) Replicación Circular: En esta topología cada servidor actúa como servidor maestro y esclavo. En este caso también es necesario usar los parámetros de configuración auto_increment_offset y auto_increment_increment para evitar valores duplicados en las columnas autoincrement.

D. Configuración de la Replicación MySQL

La configuración de la replicación en MySQL entre un servidor maestro y un servidor esclavo es una tarea sencilla. El tutorial oficial de la configuración de la replicación MySQL puede ser revisado en [23]. Básicamente, un DBA o profesional en MySQL debe realizar lo siguiente.

1) Actualizar archivos de configuración: Se debe establecer los parámetros de replicación en el archivo de configuración MySQL tanto del servidor maestro como del servidor esclavo. Luego se debe de reiniciar el servicio de MySQL para que los cambios surtan efecto.

2) Escribir y ejecutar el comando mysqlreplicate: Se debe escribir el comando mysqlreplicate pasando como mínimo los parámetros master (maestro) y slave (esclavo) y ejecutarlo en la herramienta MySQL Utilities [24].

E. Ingeniería Dirigida por Modelos

MDE se centra en la explotación de modelos como la piedra angular del proceso de desarrollo de software [8]. MDE está basado en número de principios que involucra los conceptos de modelo, metamodelo, y transformaciones de modelos para la generación automática del código durante el proceso de desarrollo de software [8], [9]. En este sentido, los modelos y las transformaciones entre dichos modelos juegan un papel central en los procesos de desarrollo, convirtiéndose en la base de este paradigma [10]. MDE nació con el lanzamiento de la Arquitectura Dirigida por Modelos (Model Driven Architecture, MDA) en el año 2001 por el consorcio OMG (Object Management Group) y con la progresiva unificación de iniciativas que usan modelos como artefacto principal en el desarrollo de software [25]. En muchas de las siglas de estas iniciativas aparecen las letras MD para referirse a la frase en inglés model driven [26], [27]. MDE, el Desarrollo Dirigido por Modelos (Model Driven Development, MDD) y el Desarrollo de Software Dirigido por Modelos (Model Driven Software Development, MDSD) son aproximaciones de desarrollo similares, mientras que MDA es una de las modalidades para implementar MDE bajo los estándares OMG [26], [27] [28].

Los DSLs desempeñan un papel cada vez más importante en MDE [11], principalmente por su simplicidad (comparado a lenguajes de propósito general como UML), y enfoque en el dominio del problema [12]. DSL no es un concepto nuevo, pero ha llegado a ser popular en los últimos años debido al enfoque de desarrollo orientado al dominio que MDE ha introducido [14]. Investigadores han mostrado mayor interés en DSLs gráficos, llamados también DSMs, porque son más expresivos que los DSLs textuales [13]. Los modelos diseñados en un DSL pueden ser validados y posteriormente usados como entrada en un proceso de transformación modelo a modelo y/o modelo a texto, la cual producen automáticamente otros modelos o artefactos basados en texto tales como código fuente o documentación [14].

F. Soporte Tecnológico para MDE

Un entorno de metamodelamiento es un medio usado para desarrollar una herramienta de modelamiento mediante la definición de su metamodelo en un meta-metamodelo predefinido [9]. Los lenguajes de metamodelado más populares son la especificación Meta Object Facility (MOF) de la OMG [29], UML[30], Eclipse Modeling Framework (EMF) [31] y MetaGME [9]. EMF, es la plataforma de metamodelado más adoptada en el ámbito de MDE [32]. Ecore es el metamodelo usado por EMF para definir metamodelos. Ecore es una implementación del lenguaje EMOF (Essential MOF), que es un subconjunto de MOF. El modelo EMOF provee de un conjunto mínimo de elementos requeridos para especificar metamodelos. Entre los principales elementos de Ecore tenemos a EClass, EReference y EAttribute.

Existen diferentes herramientas y frameworks para MDE que permiten la creación de DSMs tales como Microsoft Visualization and Modeling SDK (antes llamado DSL Tools) [33], MetaEdit+, [34], Marama [35], Graphical Editing Framework (GEF) [36], Graphical Modeling Framework (GMF) [37], GenGMF [38], KybeleGMFgen [32], Generic Modeling Environment (GME) [39], Generic Eclipse Modeling System (GEMS) [40], Eugenia [41], entre otros. GMF es uno de los frameworks más usado para generar editores gráficos para crear modelos de acuerdo al metamodelo especificado. De esta forma, el código fuente es automáticamente generado usando como datos de entrada el modelo especificado en el editor gráfico [8].

Object Constraint Language (OCL) [42] puede ser usado para definir restricciones sobre los metamodelos, estas restricciones aplican a todos los modelos que son instancias del metamodelo [9]. Epsilon Validation Language (EVL) [43] permite definir restricciones de forma similar al lenguaje OCL.

La transformación de un modelo en otro modelo se le conoce como transformación modelo a modelo (Model to Model, M2M), entre las herramientas que realizan esta tarea tenemos a ATL (ATLAS Transformation Language) [10], Operational QVT [44] y Epsilon Transformation Language (ETL) [45]. Para la generación de código existen varias tecnologías, tales como Acceleo [46], Xpand [47], MOFScript [48], JET (Java Emitter Templates) [49], Epsilon Generation Language (EGL) [50], entre otras, todas emplean el mismo principio, la creación de reglas de transformación basándose en un metamodelo. Estas reglas serán aplicadas al modelo, para generar el código fuente en el lenguaje deseado, este tipo de tecnología recibe el nombre de transformación modelo a texto (Model to Text, M2T).

La tabla 1 muestra la comparación de las herramientas y frameworks para la creación de un DSM. Tal como se puede observar en la tabla 1, se seleccionó la herramienta Eugenia del proyecto Epsilon de Eclipse [51] para el desarrollo del editor gráfico de la herramienta propuesta. La plataforma Eclipse fue elegida porque es una tecnología madura, software libre y multiplataforma. Se decidió usar Epsilon porque es una familia completa de lenguajes y herramientas maduras para desarrollar software basado en MDE. La base de Epsilon es el lenguaje Epsilon Object Language (EOL) [52]. Específicamente para la validación de modelos de replicación MySQL se eligió al lenguaje EVL y para la generación de los comandos mysqlreplicate de configuración de la replicación se eligió al lenguaje EGL.

TABLE I. COMPARACIÓN DE HERRAMIENTAS Y FRAMEWORKS PARA LA CREACIÓN DE UN EDITOR GRÁFICO DE MODELADO (DSM)

Características	**Frameworks/Herramientas**					
	GMF	***Kybele***	***Eugenia***	***GEMS***	***VMSDK***	***GME***
Generación Automática de Editor Gráfico	No	Sí	Sí	Sí	No	No
Software Libre	Sí	Sí	Sí	Sí	No	Sí
Multiplataforma	Sí	Sí	Sí	Sí	No	No
Documentación y Soporte	Sí	No	Sí	Sí	Sí	Sí
Tecnología Madura	Sí	No	Sí	No	Sí	Sí

III. ESTADO DEL ARTE

A. Herramientas para la Replicación de MySQL

1) Herramientas para el Modelado de la Replicación MySQL: Los DBA's en MySQL utilizan herramientas de diagramación tales como Microsoft Visio [6], Dia [7], Draw.io [53], Gliffy [54], ConceptDraw [55], Creately [56], FreelyDraw [57], SmartDraw [58], LucidChart [59], entre otros, para diagramar modelos de replicación de MySQL. Sin embargo, este tipo de herramientas no permiten validar automáticamente si un modelo de replicación de MySQL está libre de errores. Tampoco permiten generar a partir del modelo de replicación los comandos mysqlreplicate de configuración. En la revisión de la literatura no se encontró ninguna herramienta comercial, de software libre, o publicación académica que permita dichas funcionalidades.

2) Herramientas para la Configuración y Administración de la Replicación MySQL: Entre las principales herramientas de configuración y administración de la replicación MySQL tenemos a la herramienta MySQL Utilities [24], Openark Kit [60], MHA [61] y Percona Toolkit [62], todas ellas de línea de comandos (Command Line Interface, CLI). La herramienta gráfica MySQL Workbench [63] permite ver y editar parámetros de configuración de la replicación MySQL pero no permiten diagramar modelos de replicación MySQL.

3) Herramientas para el Monitoreo de la Replicación MySQL: Entre las principales herramientas gráficas de monitoreo tenemos a la herramienta MySQL Enterprise Monitor [64] y MySQL Performance Monitor [65].

4) Herramientas para la Visualización Gráfica de Topologías de la Replicación MySQL: Una vez configurada e implementada la replicación en MySQL es posible obtener la visualización gráfica de las topologías utilizando la herramienta MySQL Replication Diagram Tool [66] y Orchestrator [67]. Ninguna de ellas a nivel de modelado.

5) Herramientas MDE para la Replicación de MySQL: En la revisión de la literatura no se encontró ninguna herramienta desarrollada con el enfoque MDE que aborde algún aspecto relacionado con la replicación de MySQL u otro motor de base de datos relacional.

B. Herramientas MDE para el dominio de Bases de Datos

En [68] desarrollaron una herramienta gráfica para el modelado de bases de datos objeto-relacionales en Oracle 10g. La propuesta inicia con un modelo conceptual de datos (Platform Independent Model, PIM), que luego es mapeado en un modelo de base de datos objeto-relacional (Platform Specific Model, PSM) que representa el esquema de base de datos objeto-relacional, para lograr ello los autores implementaron reglas de transformación modelo a modelo en el lenguaje ATL [10]. De ser necesario el diseñador podría refinar o actualizar el modelo en la herramienta de modelado implementado en GMF. Finalmente, el código SQL que implementa el esquema modelado en Oracle 10g es automáticamente generado desde el PSM usando reglas de transformación modelo a texto con MOFScript.

En [32] se propuso KybeleGMFgen para soportar la generación automática de diagramadores para modelos EMF. La solución está basada en GMF y Eugenia. Fue aplicado un caso de estudio para el modelado de bases de datos objeto-relacionales en Oracle 10g presentado en [68].

C. Herramientas MDE para otros Dominios

En [69] presentan una herramienta gráfica para el modelado de reportes y generación de código. Para la construcción de la herramienta, los autores utilizaron Microsoft DSL Tools (ahora denominado VMSDK), seleccionaron esta herramienta debido a que el motor de reportes es una solución basada en Microsoft. Para especificar restricciones al metamodelo emplearon DSL Tools y funciones C#, y para la generación de código fuente usaron el motor de plantillas T4.

En [9] presentan una herramienta de soporte para un entorno de modelado y simulación. La herramienta está basada en EMF. Se utilizó GEMS para implementar el editor gráfico. Se desarrollaron transformaciones M2M escritas en ATL. Se definió una aplicación Java para generar el código fuente.

En [29] realizaron un editor gráfico para la construcción de módulos en sistemas de gestión del aprendizaje (LMS). El metamodelo se desarrolló con el lenguaje Ecore de EMF, usaron GMF para la construcción del editor gráfico de modelado. MOFScript fue utilizado para realizar las transformaciones modelo a texto de generación de código para el LMS seleccionado.

En [70] implementaron una herramienta para el desarrollo de aplicaciones de redes inalámbricas de sensores. Desarrollaron sus metamodelos en Ecore. Dispusieron de GenModel de EMF para crear el editor del modelo desde los metamodelos. Seleccionaron Operational QVT para desarrollar las transformaciones M2M. Fue usado el plugin de eclipse Acceleo para las transformaciones M2T.

En [8] desarrollaron una herramienta gráfica para la definición de modelos de una metodología para la construcción de sistemas multiagentes. El lenguaje Ecore de EMF se usó para especificar el metamodelo. Se empleó GMF en la implementación del editor gráfico para la creación de modelos gráficos. Los autores eligieron JET para generar automáticamente el código.

En [71] construyeron una herramienta para soportar el desarrollo basado en componentes de acuerdo a un modelo de componentes que también propusieron. La herramienta fue implementada usando GME, tanto el metamodelo como el editor gráfico de modelado fueron realizados en GME. Se desarrollaron intérpretes en C++ para la generación de código.

En [72] propusieron una herramienta de soporte para el modelo y análisis de puntos de vista de arquitectura de software. El metamodelo es definido en el lenguaje Ecore de EMF. Aprovecharon las bondades de Eugenia para la generación del editor gráfico desde un metamodelo Ecore con anotaciones. Para anotar el metamodelo Ecore con información de sintaxis concreta visual, utilizaron Emfatic.

IV. Desarrollo de la Herramienta Propuesta

Para el proceso de desarrollo de la herramienta se ha utilizado una variante simplificada del Proceso Unificado de Rational (Rational Unified Process - RUP) [73]. De las 9 disciplinas de la versión original solo se han considerado 5 y se han agregado 2, las disciplinas de Análisis del Dominio y Metamodelo del Dominio. Se han incorporado estas diferencias debido al enfoque MDE de este trabajo de investigación. En la tabla II se puede observar las disciplinas trabajadas para cada una de las 4 fases del RUP.

TABLE II. Fases y Disciplinas del Proceso de Desarrollo de la Herramienta

Fase	Disciplinas
1. Inicio	Análisis del Dominio
	Requerimientos
2. Elaboración	Metamodelo del Dominio
	Análisis y Diseño
3. Construcción	Implementación
	Pruebas
4. Transición	Despliegue

A. Fase de Inicio

1) Análisis del Dominio: El alcance de la herramienta propuesta es permitir modelar y validar las relaciones de replicación entre servidores MySQL de las topologías mostradas en la Fig. 1, y adicionalmente generar a partir de un modelo válido los comandos mysqlreplicate de configuración. En base a este alcance se realizó el análisis del dominio, mediante el estudio de la documentación oficial de la replicación de MySQL [4], experiencia propia de los autores configurando la replicación MySQL y consultando con expertos del dominio. A continuación se describe cada uno de los aspectos identificados en esta disciplina.

a) Elementos Modelables: Los elementos de la replicación MySQL que van a ser modelables con la herramienta propuesta son los siguientes:

- Servidor MySQL: Es un servidor MySQL que puede actuar como nodo maestro o esclavo. Para cada servidor MySQL se puede configurar el nombre del host y puerto. La representación visual que se espera de este elemento se ilustra en la Fig. 2.

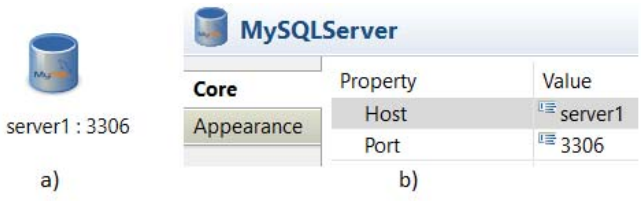


Fig. 2. a) Forma Visual de un Servidor MySQL b) Propiedades de Configuración de un Servidor MySQL

- Relación Maestro a Esclavo: Es la relación de replicación maestro a esclavo entre un par de servidores MySQL, en la que se establece el servidor maestro y el que será esclavo. La Fig. 3 muestra la representación visual que se espera de este elemento.

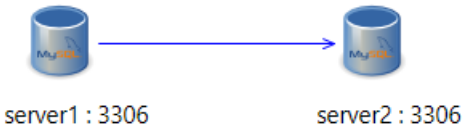


Fig. 3. Representación Visual de una Relación Maestro a Esclavo

- Relación Maestro a Maestro: En esta relación de replicación ambos servidores actúan como maestros y esclavos, en la que se establece el servidor maestro origen y el maestro destino. La representación visual que se espera de este elemento se ilustra en la Fig. 4.

Fig. 4. Representación Visual de una Relación Maestro a Maestro

b) Primitivas Conceptuales: Para representar los modelos de replicación MySQL, se identificaron las siguientes primitivas conceptuales:

- Model: Representa un modelo de replicación, la cual contiene servidores MySQL y relaciones de replicación (maestro a esclavo o maestro a maestro).
- MySQLServer: Representa un servidor MySQL.
- MasterSlaveRelationship: Representa una relación de replicación maestro a esclavo entre un par de servidores MySQL.
- MasterMasterRelationship: Representa una relación maestro a maestro entre 2 de servidores MySQL.

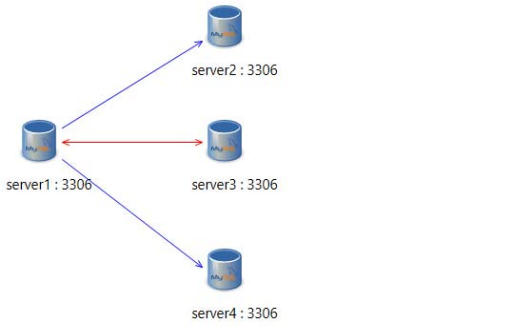
Dado que la herramienta propuesta se va a ejecutar sobre Eclipse, GMF y EMF, se heredan sus primitivas.

c) Reglas de Validación: Se identificaron las siguientes reglas que van a ser incorporadas en la herramienta para validar los modelos de replicación MySQL:

- Un servidor esclavo debe tener solo un maestro.
- Un servidor maestro puede tener cualquier número de servidores esclavos.
- Un servidor esclavo puede ser servidor maestro para otros servidores esclavos.
- Un servidor maestro o esclavo no puede tener una relación de replicación consigo mismo.
- El nombre del host de un servidor no debe ser vacío.
- El número de puerto de un servidor maestro o esclavo debe estar entre 0 y 65535.

- La unión del host y puerto de un servidor maestro o esclavo debe ser única.
- Un servidor maestro o esclavo debe ser parte de una relación de replicación.
- Solo debe existir una relación de replicación entre un par de servidores MySQL.

d) Generación de Comandos mysqlreplicate: En la Fig. 5 se puede observar un modelo de replicación MySQL que debería poder ser hecho con la herramienta propuesta, así como los comandos mysqlreplicate que debería generar la herramienta a partir del modelo.




```
mysqlreplicate --master=root@server1:3306 --slave=root@server2:3306
mysqlreplicate --master=root@server1:3306 --slave=root@server4:3306
mysqlreplicate --master=root@server1:3306 --slave=root@server3:3306
mysqlreplicate --master=root@server3:3306 --slave=root@server1:3306
```

Fig. 5. Comandos mysqlreplicate Generados a partir de un Modelo

2) Requerimientos

a) Requerimientos Funcionales: Fueron capturados con la técnica de casos de uso, y se describen en la tabla III.

TABLE III. REQUERIMIENTOS FUNCIONALES

#	Caso de Uso	Descripción
CU-01	Crear Modelo de Replicación	La herramienta permitirá crear un modelo de replicación MySQL.
CU-02	Guardar Modelo de Replicación	La herramienta permitirá guardar un modelo de replicación MySQL.
CU-03	Abrir Modelo de Replicación	La herramienta permitirá abrir un modelo de replicación MySQL.
CU-04	Agregar Servidor MySQL	La herramienta permitirá agregar un servidor a un modelo de replicación.
CU-05	Modificar Servidor MySQL	La herramienta permitirá modificar un servidor de un modelo de replicación.
CU-06	Eliminar Servidor MySQL	La herramienta permitirá eliminar un servidor de un modelo de replicación.
CU-07	Crear Relación Maestro-Esclavo	La herramienta permitirá crear una relación maestro-esclavo en el modelo.
CU-08	Eliminar Relación Maestro-Esclavo	La herramienta permitirá eliminar una relación maestro-esclavo del modelo.
CU-09	Crear Relación Maestro-Maestro	La herramienta permitirá crear una relación maestro-maestro en el modelo.
CU-10	Eliminar Relación Maestro-Maestro	La herramienta permitirá eliminar una relación maestro-maestro del modelo.
CU-11	Validar Modelo de Replicación	La herramienta permitirá validar un modelo de replicación MySQL.
CU-12	Generar Comandos mysqlreplicate	La herramienta permitirá generar los comandos mysqlreplicate del modelo.

b) Requerimientos No Funcionales: En la tabla IV se muestra los requerimientos no funcionales.

TABLE IV. REQUERIMIENTOS NO FUNCIONALES

#	Categoría	Descripción
RNF-01	Usabilidad	La herramienta debe tener una interfaz fácil de usar, en la que todos los elementos de un modelo de replicación MySQL deben ser fácilmente accesibles por el usuario. El idioma de la herramienta debe estar en inglés.
RNF-02	Escalabilidad	La herramienta debe tener la capacidad de trabajar sin problemas un modelo de replicación de hasta 200 servidores MySQL.
RNF-03	Integración	La herramienta debe ser integrada con el IDE Eclipse mediante plugins.
RNF-04	Portabilidad	La herramienta debe poder ejecutarse en sistemas Windows, Linux y MAC.

B. *Fase de Elaboración*

1) Metamodelo del Dominio: El metamodelo del dominio define la sintaxis abstracta de un DSL. El metamodelo es un artefacto de vital importancia en un desarrollo MDE ya que es el dato de entrada para la generación del editor gráfico (sintaxis concreta) de un DSL. El metamodelo contiene las primitivas conceptuales descritas en el Análisis del Dominio. La Fig. 6 muestra el modelo Ecore del metamodelo propuesto.

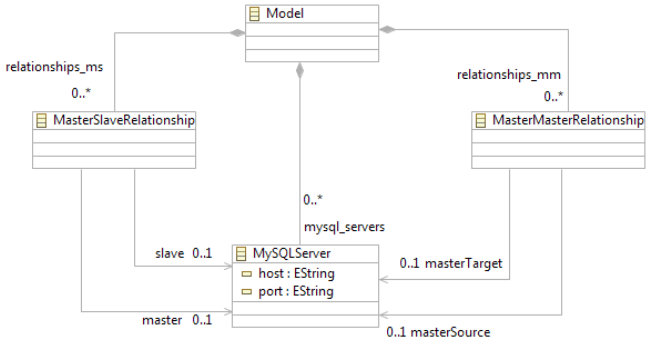


Fig. 6. Metamodelo Propuesto

2) Análisis y Diseño: En esta disciplina se detalla las vistas arquitectónicas del documento de arquitectura de software del RUP [73].

a) Vista de Casos de Uso: La Fig. 7 ilustra esta vista mediante el modelo de casos de uso.

b) Vista Lógica: Esta vista muestra la organización de la herramienta en subsistemas y cómo la funcionalidad es diseñada en el interior de la herramienta, en términos de la estructura estática (Diagrama de clases). La Fig. 8 muestra la descomposición de la herramienta en dos subsistemas:

- Editor Gráfico GMF: Subsistema para diagramar un modelo de replicación de MySQL. El código fuente java de este subsistema será automáticamente generado por la herramienta Eugenia.

- Personalizaciones: Subsistema para realizar mejoras estéticas al editor gráfico GMF, validar un modelo de replicación MySQL y para generar los comandos mysqlreplicate. En [74] se puede revisar el diagrama de clases de este subsistema.

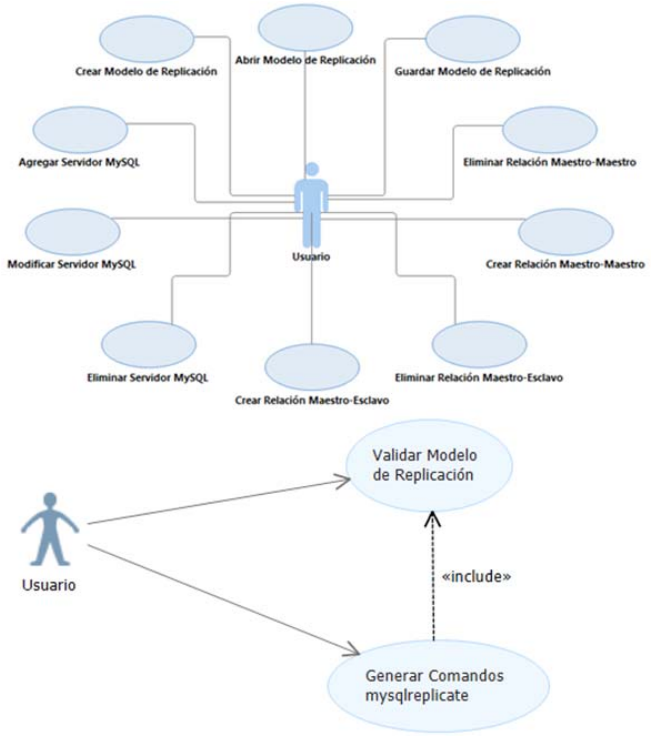


Fig. 7. Modelo de Casos de Uso

Fig. 8. Subsistemas de la Herramienta

c) Vista de Implementación: Esta vista muestra la organización del código en un diagrama de componentes. Por motivos de espacio este modelo puede ser consultado en [74] y notar la dependencia de la herramienta propuesta MySQL Replication Modeling de la plataforma Eclipse, EMF, GEF, y el runtime de GMF.

d) Vista de Despliegue: Esta vista muestra los nodos físicos mediante un diagrama de despliegue. La herramienta se desplegará como una serie de plugins del IDE de eclipse, siendo éste el único nodo físico.

e) Vista de Datos: La persistencia de los modelos de replicación MySQL se realiza en el estándar del formato XMI y no en una base de datos relacional.

C. Fase de Construcción

1) Implementación: Para la construcción de la herramienta se usó el IDE Eclipse Kepler 4.3 para el sistema operativo windows a partir de 32 bits, este IDE contiene la versión v1.1_SR1 de Epsilon, EMF, GMF y Emfatic. La Fig. 9 muestra la arquitectura tecnológica de la herramienta.

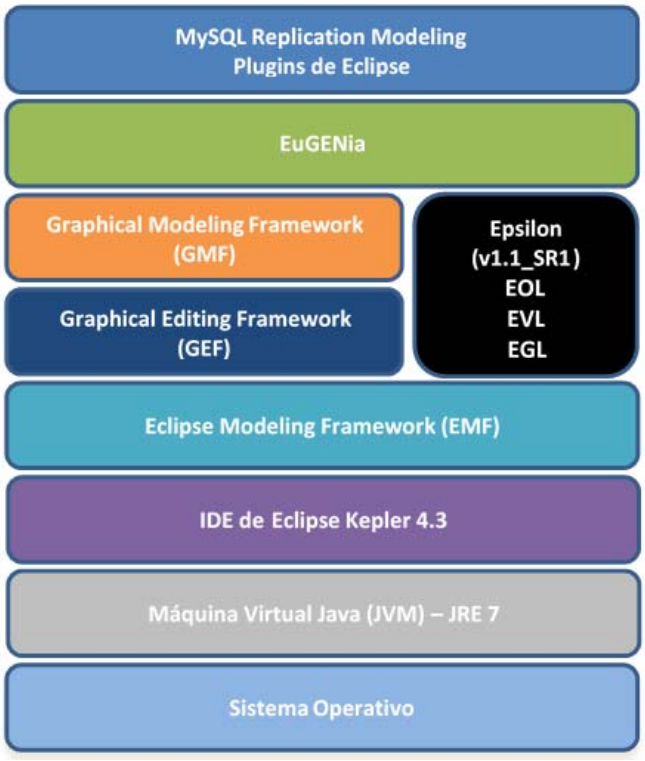


Fig. 9. Arquitectura Tecnológica de la Herramienta

A continuación se detalla cada una de las iteraciones mostradas en la tabla V.

TABLE V. ITERACIONES DE LA IMPLEMENTACIÓN

#	Iteración
1	Implementación del Metamodelo
2	Generación automática del subsistema Editor Gráfico GMF
3	Mejoras estéticas al subsistema Editor Gráfico GMF
4	Implementación de las reglas de validación para modelos de replicación MySQL del subsistema Personalizaciones
5	Implementación de la generación de comandos mysqlreplicate del subsistema Personalizaciones

a) Implementación del Metamodelo: La Fig. 10 muestra la codificación del metamodelo en un archivo Emfatic. A continuación se describe brevemente cada unos de los elementos del metamodelo.

- package: Un paquete contiene todos los nodos y enlaces del metamodelo. Un nodo es una figura que puede ser o no visible en un diagrama GMF, y los enlaces son las relaciones entre los nodos. En este trabajo un nodo representa a un servidor MySQL y

los enlaces representan las relaciones maestro-esclavo y maestro-maestro.

- class: La palabra clave "class" es usado en Emfatic para declarar una metaclase. Los atributos en Emfatic son definidos usando la palabra clave "attr" seguido por el tipo de dato y el nombre del atributo.
- @gmf.diagram: Es usado para denotar una metaclase como el elemento raíz, que viene a ser el diagrama que contiene el modelo con los nodos y enlaces que son referencias containment especificadas usando la palabra clave "val", esto es, si el diagrama es eliminado, los nodos y enlaces también lo serán.
- @gmf.node: Con esta anotación se define una metaclase como nodo. Sus parámetros especifican la etiqueta que mostrará el nodo.
- @gmf.link: Usado para definir una metaclase como enlace. Sus parámetros especifican el nodo origen y nodo destino del enlace, así como el color, tipo de decoración, entre otras características. Cada una de las meta-clases de tipo link referencian a la meta-clase MySQLServer de tipo nodo para especificar su origen y destino, estas son referencias normales y se definen con la palabra clave "ref".

metamodel.emf

```
@namespace(uri="http://www.unmsm.edu.pe/mysql_replication_modeling",
prefix="mysql_replication_modeling")
package mysql_replication_modeling;

@gmf.diagram(model.extension="rplm", diagram.extension="rpl",
onefile="true")
class Model {
  val MySQLServer[*] mysql_servers;
  val MasterSlaveRelationship[*] relationships_ms;
  val MasterMasterRelationship[*] relationships_mm;
}

@gmf.node(figure="mysql.replication.modeling.figures.MySQLServerFigure",
label="host, port", label.pattern="{0} : {1}", label.icon="false",
label.placement="external")
class MySQLServer
{
  attr String host = "server";
  attr String port = "3306";
}

@gmf.link(source="master", target="slave", style="solid", label.icon="true",
width="1", color="0,0,255", target.decoration="arrow",
source.constraint="self <> oppositeEnd")
class MasterSlaveRelationship {
  ref MySQLServer master;
  ref MySQLServer slave;
}

@gmf.link(source="masterSource", target="masterTarget", style="solid",
label.icon="true", width="1", color="255,0,0", source.decoration="arrow",
target.decoration="arrow", source.constraint="self <> oppositeEnd")
class MasterMasterRelationship {
  ref MySQLServer masterSource;
  ref MySQLServer masterTarget;
}
```

Fig. 10. Código del Metamodelo Anotado en Emfatic

b) Generación automática del subsistema Editor Gráfico GMF: Para generar el editor gráfico, Eugenia necesita como entrada el metamodelo Emfatic desarrollado en la iteración anterior. Eugenia se encargará de generar el Modelo de Herramientas (Tooling Model), el Modelo de Definición Gráfica (Graph Definition Model) y el Modelo de Mapeo (Mapping Model) requeridos por GMF. Luego Eugenia realiza la unión o merge de estos tres modelos para generar así el Modelo de Generación (Gen Model) para finalmente generar el código fuente del editor gráfico como proyectos de tipo plugin de Eclipse. En [12] se ilustra el proceso de generación de un editor gráfico con Eugenia. El plugin .diagram generado por Eugenia, contiene la lógica principal del editor gráfico. La funcionalidad y estética de la herramienta en esta iteración es muy limitada.

c) Mejoras estéticas al subsistema Editor Gráfico GMF: Las anotaciones disponibles en Emfatic son limitadas y no permiten obtener un editor gráfico GMF con características complejas, es por ello, Eugenia permite a los ingenieros de software implementar tres modelos textuales de personalización (ECore2GMF.eol, FixGenModel.eol y FixGMFGen.eol) descritos en el lenguaje EOL. Para personalizar el editor gráfico se consideró codificar el archivo ECore2GMF.eol. Este archivo se ejecuta cada vez que Eugenia genera los modelos .gmfgraph, .gmftool y .gmfmap, por lo que se emplea para personalizar dichos modelos. Otro punto importante es que los nodos que representan los servidores MySQL se muestren con figuras. Para lograr esto, se genera el plugin Figures desde el archivo .gmfgraph. Para cambiar los iconos de la paleta de herramientas, se reemplaza los archivos .gif creados en el plugin .edit. Otro cambio fue modificar las etiquetas de los archivos .properties de los plugins generados y reemplazarlos con valores representativos del dominio.

d) Implementación de las reglas de validación para modelos de replicación MySQL del subsistema Personalizaciones: Se desarrollaron las reglas de validación al metamodelo usando el lenguaje EVL. Por cuestiones de espacio, el código EVL puede ser consultado en el trabajo de tesis de maestría de esta investigación en [74].

e) Implementación de la generación de comandos mysqlreplicate del subsistema Personalizaciones: Las transformaciones M2T se desarrollaron usando el lenguaje EGL y el código EGL puede ser consultado en [74].

2) Pruebas: La herramienta propuesta es un editor gráfico GMF que se integra en el IDE Eclipse por medio de plugins. GMF es un framework maduro, probado, correcto y funcional. Las pruebas realizadas por los autores cumplieron satisfactoriamente la funcionalidad requerida para las topologías de la Fig 1. En la sección VI se muestran los experimentos y resultados del uso de la herramienta por profesionales con experiencia en la replicación MySQL.

D. Fase de Transición

1) Despliegue: La herramienta consta de varios plugins para el IDE Eclipse. El archivo .jar de cada plugin se debe copiar en la carpeta dropins del IDE Eclipse.

V. EJEMPLO ILUSTRATIVO DE LA HERRAMIENTA

La Fig. 11 muestra la validación automática de un modelo de replicación MySQL con errores hecho con la herramienta propuesta. Se puede observar en la Fig. 11 la validación de las reglas identificadas en la disciplina de Análisis del Dominio: (i) un servidor debe tener solo un maestro (ii) el nombre del host de un servidor no debe ser vacío (iii) el rango permitido del número del puerto (iv) la

unión del host y puerto debe ser única (v) un servidor debe ser parte de una relación de replicación (vi) solo debe existir una sola relación de replicación entre 2 servidores MySQL (vii) un servidor maestro puede tener varios esclavos (viii) un servidor esclavo puede ser servidor maestro para otros esclavos y (ix) la herramienta no permite modelar que un servidor MySQL tenga una relación de replicación consigo mismo. Se puede notar en la Fig. 11 que la herramienta resalta los servidores MySQL y las relaciones de replicación que presentan errores en el modelo para una rápida identificación y corrección por parte del usuario.

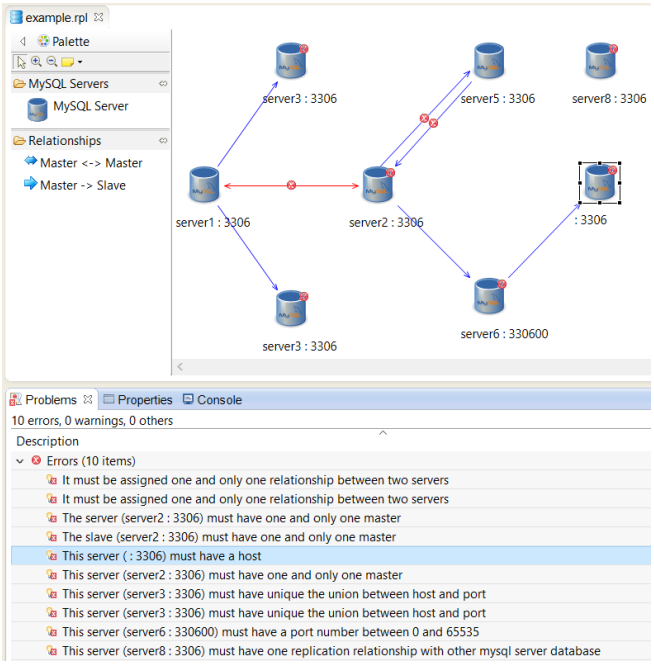


Fig. 11. Validación Automática de un Modelo de Replicación MySQL con Errores

Una vez que el usuario haya corregido los errores del modelo, éste puede volver a la ejecutar en la herramienta la opción de validación para asegurarse que el modelo está libre de errores. La Fig. 12 muestra esta validación.

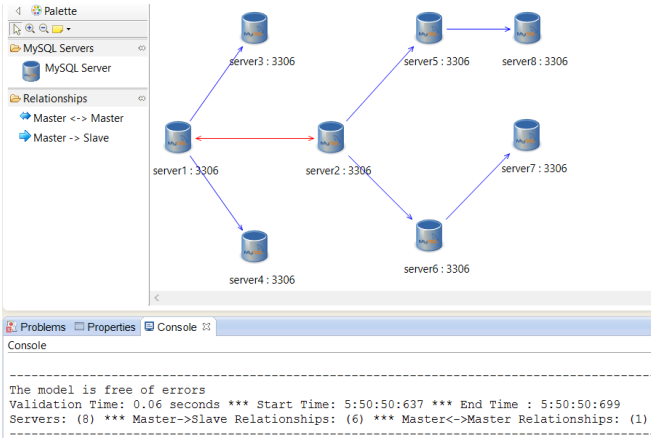


Fig. 12. Validación Automática de un Modelo de Replicación MySQL sin Errores

La Fig. 13 muestra los comandos mysqlreplicate generados a partir del modelo de la Fig. 12.

```
#Master to Slave relationships
mysqlreplicate --master=root@server1:3306 --slave=root@server4:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server1:3306 --slave=root@server3:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server2:3306 --slave=root@server6:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server2:3306 --slave=root@server5:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server5:3306 --slave=root@server8:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server6:3306 --slave=root@server7:3306 --rpl-user=rpl:rpl -vv --pedantic

#Master to Master relationships
mysqlreplicate --master=root@server1:3306 --slave=root@server2:3306 --rpl-user=rpl:rpl -vv --pedantic
mysqlreplicate --master=root@server2:3306 --slave=root@server1:3306 --rpl-user=rpl:rpl -vv --pedantic
```

Fig. 13. Comandos mysqlreplicate Generados por la Herramienta

VI. Experimentos y Resultados

Los experimentos se llevaron a cabo con la participación de diez (10) profesionales con experiencia en la replicación MySQL. A cada participante se le asignó una PC con Windows 7 Home Premium, procesador Intel Core i5 2.3 GHz, 4GB de RAM, y la instalación de la herramienta propuesta denominada MySQL Replication Modeling y Microsoft Visio 2013. Antes de iniciar los experimentos, los participantes fueron capacitados en el uso de ambas herramientas y en las mediciones de tiempo a ser realizadas. Los participantes confirmaron el cumplimiento de la funcionalidad de la herramienta propuesta.

Los experimentos miden el tiempo empleado en:

- La corrección por parte del participante, de un modelo de replicación MySQL diseñado en Microsoft Visio 2013. Esta tarea implica la identificación manual de los errores del modelo.
- La corrección por parte del participante, de un modelo de replicación MySQL diseñado en MySQL Replication Modeling. Esta tarea implica la identificación automática de los errores del modelo y confirmación automática cuando un modelo es válido.

También se midió el tiempo empleado en:

- La escritura manual de los comandos mysqlreplicate de un modelo de replicación válido diseñado en Microsoft Visio 2013.
- La generación automática de los comandos mysqlreplicate a partir de un modelo de replicación válido diseñado en MySQL Replication Modeling.

Se definió cinco instancias de prueba conformadas por el número de servidores en un modelo de replicación MySQL. Se trabajó con modelos de 5, 10, 15, 20 y 25 servidores.

La tabla VI muestra los resultados del tiempo promedio empleado por los participantes en la corrección de errores de un modelo de replicación MySQL, al usar Microsoft Visio 2013 y MySQL Replication Modeling. En los resultados de la tabla VI se puede observar que el porcentaje de error es 0% al usar ambas herramientas. Sin embargo, para la corrección de errores de un modelo de replicación con 25 servidores, se demuestra que al usar la herramienta propuesta MySQL Replication Modeling, el tiempo empleado se reduce en 87.43%, comparado con el uso de la herramienta Microsoft Visio 2013. Se puede notar en la tabla VI que a medida que el número de servidores MySQL del modelo de

replicación es mayor, la reducción del tiempo aumenta. La Fig. 14 ilustra usando los datos de la tabla VI, el comportamiento del tiempo empleado en la corrección de errores de un modelo de replicación MySQL, al usar la herramienta propuesta MySQL Replication Modeling, en comparación con la herramienta Microsoft Visio 2013.

TABLE VI. TIEMPO EMPLEADO EN LA CORRECCIÓN DE ERRORES DE UN MODELO DE REPLICACIÓN MYSQL POR MYSQL REPLICATION MODELING Y MICROSOFT VISIO 2013

Servidores del Modelo	Microsoft Visio 2013		MySQL Replication Modeling		Reducción de Tiempo	
	Segundos	*% Error*	*Segundos*	*% Error*	*%*	*Minutos*
5	92.20	0	34.40	0	62.69	0.96
10	193.00	0	49.30	0	74.46	2.40
15	331.00	0	51.30	0	84.50	4.66
20	421.10	0	53.70	0	87.25	6.12
25	595.70	0	74.90	0	87.43	8.68

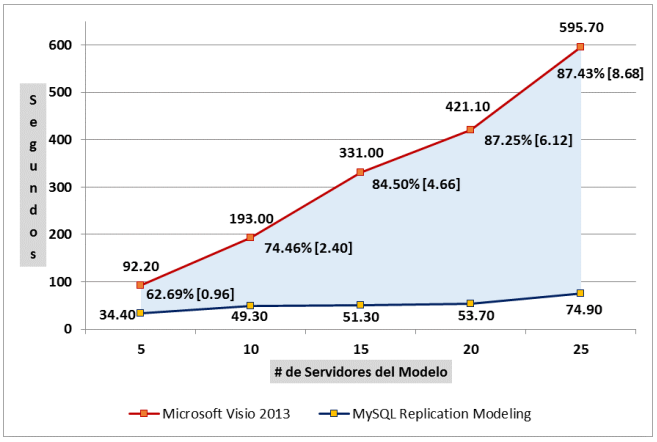


Fig. 14. Comportamiento del Tiempo en la Corrección de un Modelo de Replicación de MySQL Replication Modeling y Microsoft Visio 2013

La tabla VII muestra los resultados del tiempo promedio empleado por los participantes en la generación de comandos mysqlreplicate a partir de un modelo de replicación MySQL válido, al usar Microsoft Visio 2013 (escritura manual) y MySQL Replication Modeling (generación automática). En los resultados de la tabla VII se puede observar que el porcentaje de error es 0% al usar ambas herramientas. Sin embargo, para la generación de comandos mysqlreplicate de configuración a partir de un modelo de replicación con 25 servidores, se demuestra que al usar la herramienta propuesta MySQL Replication Modeling, el tiempo empleado se reduce en 99.78%, comparado con el uso de la herramienta Microsoft Visio 2013. Se puede notar también en la tabla VII que a medida que el número de servidores MySQL del modelo de replicación es mayor, la reducción del tiempo aumenta. La Fig. 15 ilustra usando los datos de la tabla VII, el comportamiento del tiempo empleado en la generación de comandos mysqlreplicate de configuración a partir de un modelo de replicación válido, al usar MySQL Replication Modeling, en comparación con la herramienta Microsoft Visio 2013.

TABLE VII. TIEMPO EMPLEADO EN LA GENERACIÓN DE COMANDOS MYSQLREPLICATE POR MYSQL REPLICATION MODELING Y MICROSOFT VISIO 2013

Servidores del Modelo	Microsoft Visio 2013		MySQL Replication Modeling		Reducción de Tiempo	
	Segundos	*% Error*	*Segundos*	*% Error*	*%*	*Minutos*
5	82.00	0	0.79	0	99.04	1.35
10	161.50	0	0.81	0	99.50	2.68
15	241.00	0	0.83	0	99.66	4.00
20	294.20	0	0.86	0	99.71	4.89
25	397.60	0	0.88	0	99.78	6.61

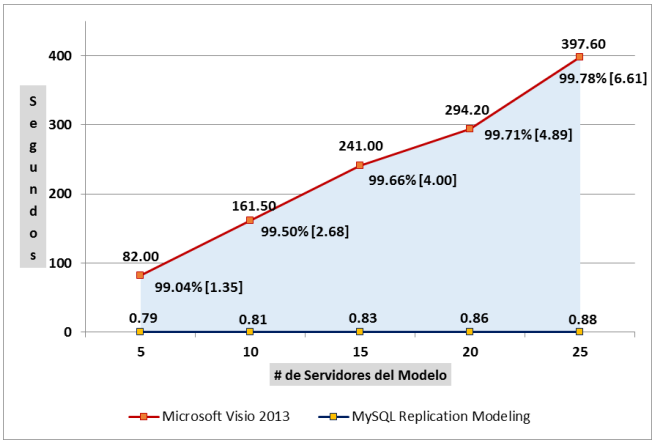


Fig. 15. Comportamiento del Tiempo en la Generación de Comandos mysqlreplicate de MySQL Replication Modeling y Microsoft Visio 2013

Se realizaron validaciones para asegurar cumplir también con los requisitos no funcionales:

- RNF-01 Usabilidad: La herramienta hereda la interfaz de usuario que la plataforma Eclipse y el framework GMF proveen. La interfaz es fácil de usar y los elementos que el usuario especifica en el modelo son fácilmente accesibles. Los participantes respondieron a la pregunta ¿Es fácil de usar la herramienta MySQL Replication Modeling?. La escala de las respuestas es (1) Totalmente en desacuerdo (2) En desacuerdo (3) Medianamente de acuerdo (4) De acuerdo y (5) Totalmente de acuerdo. Se obtuvo un promedio de 4.6 de un total de 5 puntos, confirmando la facilidad de uso de la herramienta propuesta.
- RNF-02 Escalabilidad: Se creó un modelo de replicación con 200 servidores MySQL y la herramienta trabajó sin problemas. En [14] se realizaron pruebas con 2000 elementos y 1999 enlaces y el editor GMF trabajó sin problemas.
- RNF-03 Integración: La herramienta está integrada con Eclipse. Eclipse con EMF y GMF soportan y simplifican la aplicación de enfoques MDE.
- RNF-04 Portabilidad: La herramienta corre sobre la máquina virtual de Java, por lo tanto se puede ejecutar en sistemas Windows, Linux y MAC.

VII. Conclusiones y Trabajos futuros

MDE es un enfoque de la ingeniería de software que mediante el uso de frameworks, herramientas y lenguajes maduros, es una técnica adecuada para acelerar el desarrollo de editores gráficos de modelado conocidos en la literatura como DSMs, permitiendo validar los modelos y generar código a partir de éstos.

En este artículo se presenta el proceso de desarrollo de una herramienta basada en MDE bajo la plataforma Eclipse para la validación automática de modelos de replicación de MySQL. La herramienta propuesta es también capaz de generar los comandos mysqlreplicate de configuración a partir de un modelo de replicación válido.

Los resultados de los experimentos para la corrección de errores de modelos de replicación con 25 servidores demuestran que al usar la herramienta propuesta el tiempo empleado es reducido en más del 87% comparado con el uso de la herramienta Microsoft Visio 2013. La reducción de tiempo será cada vez mayor cuando se incremente el número de servidores MySQL del modelo de replicación.

Como trabajo futuro se pretende poder modelar la gran mayoría de los parámetros de replicación configurables en el archivo de configuración de un servidor MySQL. Otro trabajo futuro es migrar la herramienta a un ambiente web manteniendo el enfoque MDE. También se puede considerar modelar y validar modelos de replicación para otros sistemas gestores de bases de datos. Otro trabajo futuro es integrar las validaciones del metamodelo con una base de reglas de un sistema experto.

Referencias

[1] C. Plattner, and G. Alonso, "Ganymed: scalable replication for transactional web applications," in Proceedings of the 5th ACM/IFIP/USEINX International Conference on Middleware, 2004, pp. 155–174.

[2] M. Ahmed, M. Uddin, S. Azad, and S. Hasseb, "MySQL performance analysis on a limited resource server: Fedora vs. Ubuntu Linux," in Proceedings of the 2010 Spring Simulation Multiconference, Society for Computer Simulation International, Paper 99.

[3] Y. Li, J. Jun, and L. Ling, "Study of Real-time Database Based on Common Information Model" in 2013 Third International Conference on Intelligent System Design and Engineering Applications (ISDEA), pp.1312–1315.

[4] Oracle, "MySQL 5.6 Replication: An Introduction", White paper, 2014. [Online]. Available: https://www.mysql.com/why-mysql/white-papers/-mysql-replication-introduction

[5] Oracle, "MySQL Replication: High Availability - Building a Self-Healing Replication Topology", White paper, 2013. [Online]. Available: https://www.mysql.com/why-mysql/white-papers/mysql-replication-high-availability

[6] Microsoft Visio 2013, Microsoft. Último acceso: Mayo de 2015. [Online]. Available: https://products.office.com/en-us/visio/microsoft-visio-2013-top-features-diagram-software

[7] Dia. Último acceso: Mayo de 2015. [Online]. Available: https://wiki.gnome.org/Apps/Dia

[8] J. Gascueña, E. Navarro, and A.Fernández-Caballero, ""Model-driven engineering techniques for the development of multi-agent systems," Engineering Applications of Artificial Intelligence 25, no. 1, 2012, pp.159–173.

[9] D. Cetinkaya, A. Verbraeck, and M. Seck, "MDD4MS: a model driven development framework for modeling and simulation," in Proceedings of the 2011 Summer Computer Simulation Conference, Society for Modeling & Simulation International, pp.113–121.

[10] A. Jimenez, "Incorporando la gestión de la trazabilidad en un entorno de desarrollo de transformaciones de modelos dirigido por modelos," Ph.D. dissertation, Rey Juan Carlos Univ., Madrid, Spain, 2012.

[11] D. Kolovos, A. García-Domínguez, L. Rose, and R. Paige. "Eugenia: towards disciplined and automated development of GMF-based graphical model editors" in Software & Systems Modeling, 2015, pp. 1–27.

[12] D. Kolovos, L. Rose, R. Paige, and F. Polack. "Raising the level of abstraction in the development of GMF-based graphical model editors," in Proceedings of the 2009 ICSE Workshop on Modeling in Software Engineering, pp. 13–19.

[13] S. Temate, L. Broto, A. Tchana, and D. Hagimont. "A high level approach for generating model's graphical editors," in Proceedings of the 2011 Eighth International Conference on Information Technology: New Generations, pp. 743–749.

[14] A. Spanou, "Reflective Diagram-Based Model Editing," M.S. thesis, Univ. of York, York, United Kingdom, 2012.

[15] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso, "Database replication techniques: a three parameter classification," in Proceedings of the 19th IEEE Symposium on Reliable Distributed Systems, 2000, pp.206–215.

[16] R. Al-Ekram, and R. Holt, "OSSR: optimal single site replication," in Proceedings of the International Symposium on Parallel and Distributed Processing with Applications, 2010, pp. 433–441.

[17] M. Araújo, "Database replication in large scale systems," M.S. thesis, Univ. of Minho, Braga, Portugal, 2011.

[18] S. Elnikety, F. Pedone, and W. Zwaenepoel, "Database replication using generalized snapshot isolation," in Proceedings of the 24th IEEE Symposium on Reliable Distributed Systems, 2005, pp. 73–84.

[19] E. Cecchet, G. Candea, and A. Ailamaki, "Middleware-based database replication: the gaps between theory and practice," in Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 739–752.

[20] MySQL 3.23, 4.0, 4.1 Reference Manual :: 14 Replication :: 14.1 Introduction to Replication. Último acceso: Mayo de 2015. [Online]. Available: http://dev.mysql.com/doc/refman/4.1/en/replication-intro.html

[21] MySQL 3.23, 4.0, 4.1 Reference Manual :: C MySQL Release Notes :: C.3 Changes in Release 3.23.x (Lifecycle Support Ended) :: C.3.46 Changes in Release 3.23.15 (08 May 2000). Último acceso: Mayo de 2015. [Online]. Available: http://dev.mysql.com/doc/refman/4.1/en/news-3-23-15.html

[22] MySQL 5.6 Reference Manual :: 17 Replication. Último acceso: Mayo de 2015. [Online]. Available: http://dev.mysql.com/doc/refman/5.6/en/replication.html

[23] Oracle, "MySQL 5.6 Replication: Enhancing Scalability and Availability – A Tutorial", White paper, 2014. [Online]. Available: http://www.mysql.com/why-mysql/white-papers/mysql-replication-tutorial

[24] MySQL Utilities, Oracle. Último acceso: Mayo de 2015. [Online]. Available: https://dev.mysql.com/downloads/utilities

[25] D. Ameller David, "Considering non-functional requirements in model-driven engineering," M.S. thesis, Polytechnic Univ. of Catalunya, Barcelona, Spain, 2009.

[26] M. Voelter. Best practices for DSLs and model-driven development. Journal of Object Technology, 2009, 8(6), pp. 79–102.

[27] J. Quintero, and J. Duitama. Reflexiones acerca de la adopción de enfoques centrados en modelos en el desarrollo de software. Ingeniería y Universidad, 2011, 15(1), pp. 219–243.

[28] J. Whittle, J. Hutchinson, and M. Rouncefield. The state of practice in model-driven engineering. Software IEEE, 2014, 31(3), pp.79–85.

[29] MetaObject Facility (MOF). Último acceso: Mayo de 2015. [Online]. Available: http://www.omg.org/mof

[30] Unified Modeling Language (UML). Último acceso: Mayo de 2015. [Online]. Available: http://www.uml.org

[31] Eclipse Modeling Framework (EMF). Último acceso: Mayo de 2015. [Online]. Available: https://www.eclipse.org/modeling/emf

[32] A. Jiménez, J. Vara, V. Bollati, and E. Marcos. "Mejorando el nivel de automatización en el desarrollo dirigido por modelos de editores gráficos," in Actas de Talleres de las XV Jornadas de Ingeniería del Software y Bases de Datos, 2010, pp. 29–37.

[33] Visualization and Modeling SDK, Microsoft. Último acceso: Mayo de 2015. [Online]. Available: https://msdn.microsoft.com/en-us/library/bb126259.aspx

[34] MetaEdit+, MetaCase. Último acceso: Mayo de 2015. [Online]. Available: http://www.metacase.com/products.html

[35] J. Grundy, J. Hosking, J. Huh, and K. Li, "Marama: an eclipse meta-toolset for generating multi-view environments," in Proceedings of the 30th international conference on Software engineering, 2008, pp. 819–822.

[36] Graphical Editing Framework (GEF). Último acceso: Mayo de 2015. [Online]. Available: http://eclipse.org/gef

[37] Graphical Modeling Framework (GMF). Último acceso: Mayo de 2015. [Online]. Available: http://eclipse.org/gmf-tooling

[38] E. Schnepel. "GenGMF: efficient editor development for large meta models using the graphical modeling framework," in Model Driven Software Engineer-ing-Transformations and Tools, 2008.

[39] Generic Modeling Environment (GME). Último acceso: Mayo de 2015. [Online]. Available: http://www.isis.vanderbilt.edu/projects/GME

[40] Generic Eclipse Modeling System (GEMS). Último acceso: Mayo de 2015. [Online]. Available: http://eclipse.org/gmt/gems

[41] D. Kolovos, L. Rose, S. Abid, R. Paige, F. Polack, and G. Botterweck. "Taming EMF and GMF using model transformation," in Model Driven Engineering Languages and Systems, 2010, pp. 211–225.

[42] Object Constraint Language (OCL). Último acceso: Mayo de 2015. [Online]. Available: http://www.omg.org/spec/OCL

[43] D. Kolovos, R. Paige, and F. Polack. "On the evolution of OCL for capturing structural constraints in modelling languages," in Rigorous Methods for Software Construction and Analysis, 2009, pp. 204–218.

[44] C. Montenegro, P. Gaona, J. Cueva, and O. Sanjuán. Aplicación de ingeniería dirigida por modelos (MDA), para la construcción de una herramienta de modelado de dominio específico (DSM) y la creación de módulos en sistemas de gestión de aprendizaje (LMS) independientes de la plataforma. DYNA, 2011, 78(169), pp. 43–52.

[45] D. Kolovos, R. Paige, and F. Polack. "The epsilon transformation language," in Theory and practice of model transformations, 2008, pp. 46–60.

[46] Acceleo. Último acceso: Mayo de 2015. [Online]. Available: http://www.eclipse.org/acceleo

[47] Xpand. Último acceso: Mayo de 2015. [Online]. Available: http://wiki.eclipse.org/Xpand

[48] MOFScript. Último acceso: Mayo de 2015. [Online]. Available: http://www.eclipse.org/gmt/mofscript

[49] JET. Último acceso: Mayo de 2015. [Online]. Available: http://www.eclipse.org/modeling/m2t/?project=jet

[50] L. Rose, R. Paige, D. Kolovos, and F. Polack. "The epsilon generation language," in Model Driven Architecture–Foundations and Applications, 2008, pp. 1–16.

[51] Epsilon. Último acceso: Mayo de 2015. [Online]. Available: http://eclipse.org/epsilon

[52] R. Paige, D. Kolovos, L. Rose, N. Drivalos, and F. Polack. "The design of a conceptual framework and technical infrastructure for model management language engineering," in Proceedings of the 2009 14th IEEE International Conference on Engineering of Complex Computer Systems, pp. 162–171.

[53] Draw.io, JGraph Ltd. Último acceso: Mayo de 2015. [Online]. Available: https://www.draw.io

[54] Gliffy. Último acceso: Mayo de 2015. [Online]. Available: https://www.gliffy.com

[55] ConceptDraw, CS Odessa LLC. Último acceso Mayo de 2015. [Online]. Available: http://www.conceptdraw.com/products/drawing-tool

[56] Creately, Cinergix Pty. Ltd. Último acceso: Mayo de 2015. [Online]. Available: http://creately.com

[57] FreelyDraw. Último acceso: Mayo de 2015. [Online]. Available: http://freelydraw.com/home.html

[58] SmartDraw, SmartDraw, LLC. Último acceso: Mayo de 2015. [Online]. Available: http://www.smartdraw.com

[59] LucidChart, Lucid Software Inc. Último acceso: Mayo de 2015. [Online]. Available: https://www.lucidchart.com

[60] Openark Kit, S. Noach. Último acceso: Mayo de 2015. [Online]. Available: http://code.openark.org/forge/openark-kit

[61] MHA, Y. Matsunobu. Último acceso: Mayo de 2015. [Online]. Available: https://code.google.com/p/mysql-master-ha

[62] Percona Toolkit for MySQL, Percona LLC. Último acceso: Mayo de 2015. [Online]. Available: http://www.percona.com/software/percona-toolkit

[63] MySQL Workbench, Oracle. Último acceso: Agosto de 2015. [Online]. Available: https://www.mysql.com/products/workbench/features.html

[64] Oracle, "MySQL Enterprise Monitor", 2013. Último acceso: Mayo de 2015. [Online]. Available: http://www.mysql.com/why-mysql/presentations/mysql-enterprise-monitor

[65] FromDual, "MySQL Performance Monitor". Último acceso: Mayo de 2015. [Online]. Available: http://www.fromdual.com/mysql-performance-monitor

[66] MySQL Replication Diagram Tool, S. McCartney. Último acceso: Mayo de 2015. [Online]. Available: https://github.com/simonmcc/mysql-replication

[67] Orchestrator, S. Noach. Último acceso: Mayo de 2015. [Online]. Available: https://github.com/outbrain/orchestrator

[68] J. Vara, B. Vela, V. Bollati, and E. Marcos. "Supporting model-driven development of object-relational database schemas: a case study," in Proceedings of the 2nd International Conference on Theory and Practice of Model Transformations, pp. 181–196.

[69] R. Dantra, J. Grundy, and J. Hosking. "A domain-specific visual language for report writing using Microsoft DSL tools," in IEEE Symposium on Visual Languages and Human-Centric Computing, 2009, pp. 15–22.

[70] T. Rodrigues, P. Dantas, F. Delicato, P. Pires, L. Pirmez, T. Batista, C. Miceli, and A. Zomaya. "Model-driven development of wireless sensor network applications," in International Conference on Embedded and Ubiquitous Computing, 2011, pp. 11–18.

[71] K. Lau, and C. Tran. "X-MAN: An MDE tool for component-based system development," in 38th Conference on Software Engineering and Advanced Applications, 2012, pp. 158–165.

[72] B. Tekinerdogan, and E. Demirli. "Evaluation framework for software architecture viewpoint languages," in International ACM Sigsoft conference on Quality of software architectures, 2013, pp. 89–98.

[73] P. Kruchten, The rational unified process: an introduction, 3rd ed. Addison-Wesley Professional, 2004.

[74] E. Bautista, "Herramienta para el Modelado de la Replicación de MySQL Basada en la Ingeniería Dirigida por Modelos," M.S. thesis, Univ. Nacional Mayor de San Marcos, Lima, Perú, 2014. [Online]. Available: https://drive.google.com/file/d/0B06ZVhhtvd0dV1VzU0Q3MFhqSE0/view?usp=sharing