Análisis y uso de los frameworks de Eclipse para la definición de DSLs

Andino Luciano Omar
Ruiz Germán Esteban

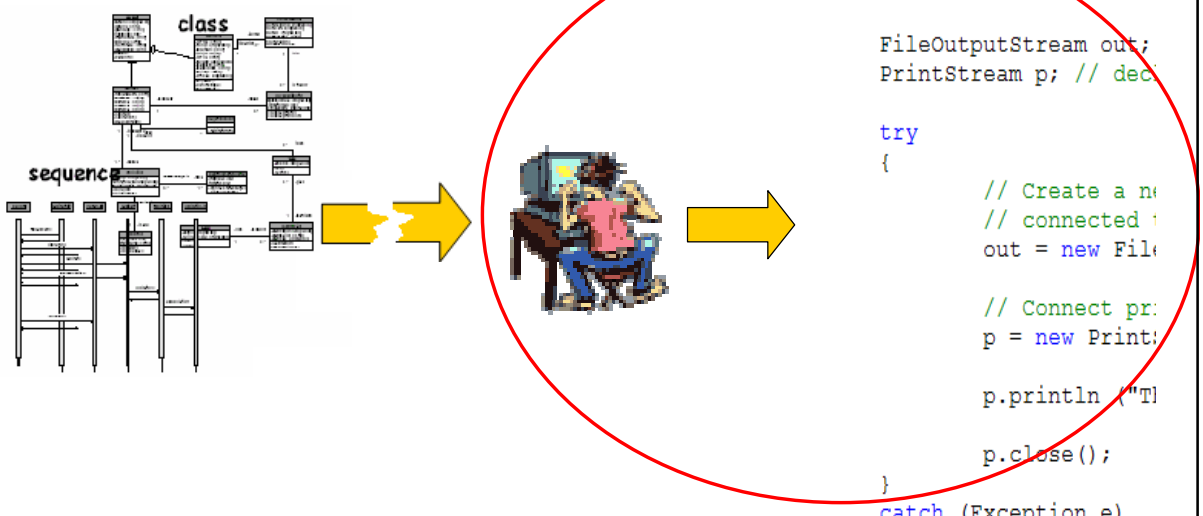
Director: Lic. Luis Mariano Bibbo
Co-Directora: Dra. Claudia Pons

Facultad de Informática
Universidad Nacional de La Plata

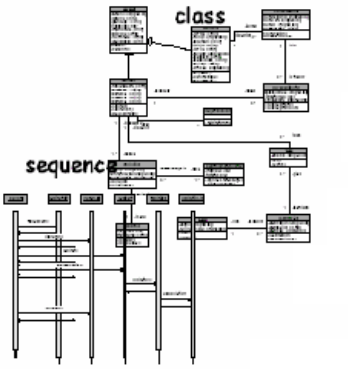
Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

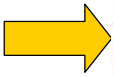
Introducción



Introducción



```
static void main(Strin

 FileOutputStream out;
 PrintStream p; // dec

 try
 {
         // Create a n
         // connected
         out = new Fil

         // Connect pr
         p = new Print

         p.println ("T

         p.close();
 }
 catch (Exception e)
```



Introducción - Conceptos

DSM

Es una metodología orientada a crear modelos para un dominio, utilizando un lenguaje enfocado y especializado para el mismo.

DSL

Es un lenguaje creado de tal manera que permite especificar la solución utilizando directamente conceptos del dominio.

Objetivo del Trabajo

- Estudiar los lenguajes específicos de dominio. En particular como se debe formalizar su definición.
- Analizar la especificación de DSLs a través del metamodelado, estudiando los estándares definidos para este propósito por la OMG.
- Analizar los frameworks ofrecidos por Eclipse para la definición de DSLs, como EMF, GMF, TMF.
- Desarrollar un caso de estudio planteando un dominio particular, especificando el DSL correspondiente, e implementándolo con los plugins de Eclipse.

Introducción

- DSLs vs LPG
  - Los lenguajes de propósito general sirven para abordar cualquier tipo de problemas, sin importar la naturaleza de los mismo.
  - Los lenguajes específicos de dominio se pueden utilizar sólo para problemas del contexto para el cual fueron creados.

Introducción

- DSL vs Frameworks
  - Los frameworks y APIs agregan conceptos mas complejos y abstractos sobre un lenguaje existente
  - Los DSLs ofrecen un conjunto de conceptos complejos y abstractos, y además una nueva sintaxis. Es decir que se crea un nuevo lenguaje.

Introducción

- MDD vs DSM
  - MDD
    - Intenta mejorar el proceso de construcción de software basándose en un proceso guiado por modelos y soportado por herramientas.
    - Los modelos se van generando desde los mas abstractos a los mas concretos aplicando transformaciones, hasta llegar al código fuente.
  - DSM
    - Utiliza conceptos del domino, Modelo, Metamodelo y Meta metamodelo como MDD.
    - Propone una forma de automatización en el ciclo de vida del Software.

Introducción

- Herramientas para crear soluciones DSM

- Software Factories de Microsoft

- MetaEdit+ de Metacase

- Eclipse Modeling Project

DSL - Beneficios

- ✓ Posee notaciones apropiadas para el dominio.
- ✓ Sirve como documentación para los expertos del dominio.
- ✓ Disminuye la posibilidad de errores en la codificación.
- ✓ Mejora la productividad.
- ✓ Ayuda a mantener los modelos actualizados

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

Lenguaje Específico de Dominio

Un DSL es un lenguaje que permite expresar soluciones a problemas de un dominio de específico.

Make

LaTeX

DSL - Formalismos

- Gramáticas:
  - libres de contexto
  - de atributos
  - de grafos
- Perfiles de UML
- Metamodelado

DSL - Construcción

- Etapas para la construcción de un DSL utilizando metamodelado:
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

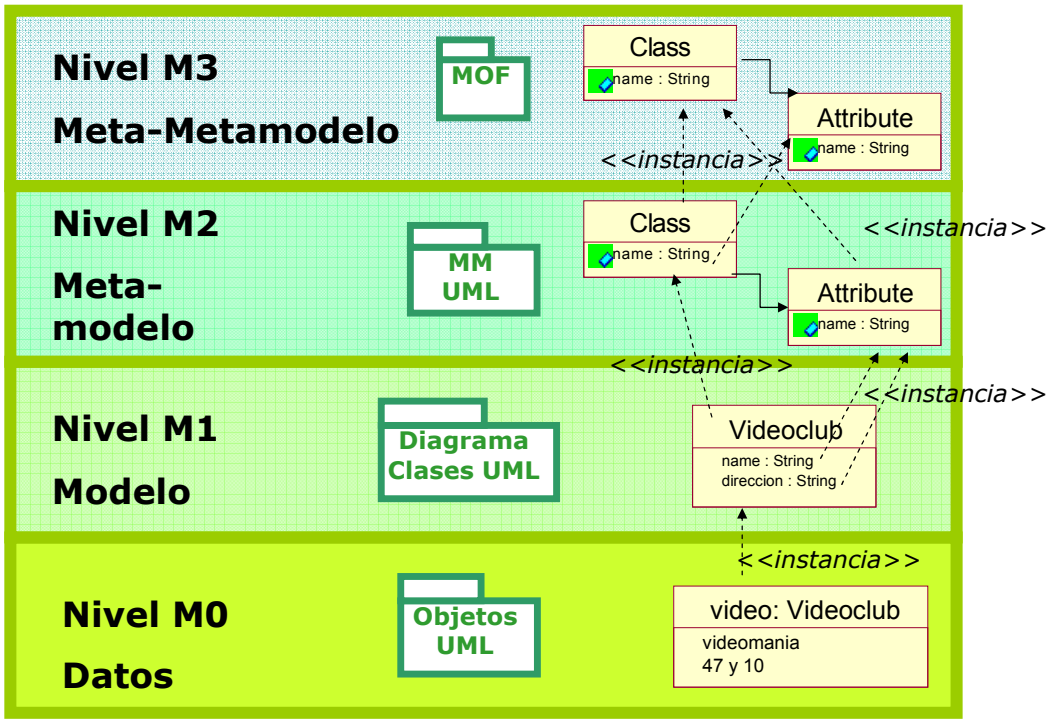
Sintaxis Abstracta

- La sintaxis abstracta describe los conceptos y las relaciones existentes entre ellos.
- Características de los conceptos:
  - Son conocidos y usados
  - Tienen establecida una semántica
  - Establecen un mapeo natural con el problema que se quiere resolver
- Además, define reglas que determinan si un modelo escrito en ese lenguaje es un modelo válido.

Sintaxis Abstracta - Modelado

- Arquitectura de cuatro capas definida por la OMG para el modelado
  - Nivel M0: instancias
  - Nivel M1: modelo
  - Nivel M2: metamodelo
  - Nivel M3: meta-metamodelo

Sintaxis Abstracta - Modelado



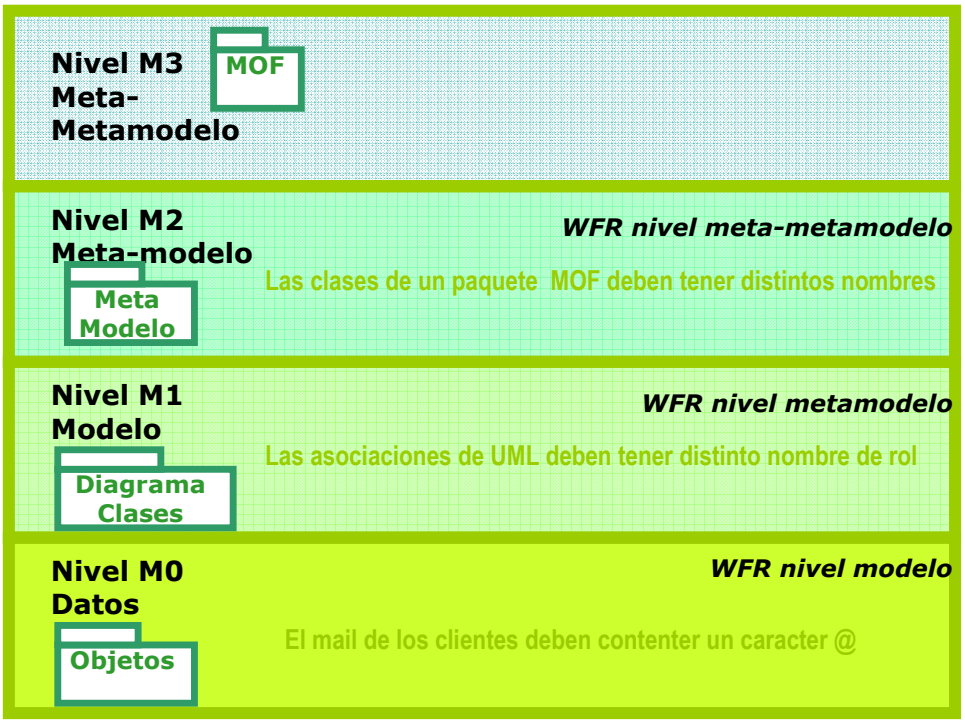
Sintaxis Abstracta – WFR OCL

OCL permite especificar diferentes tipos de restricciones:

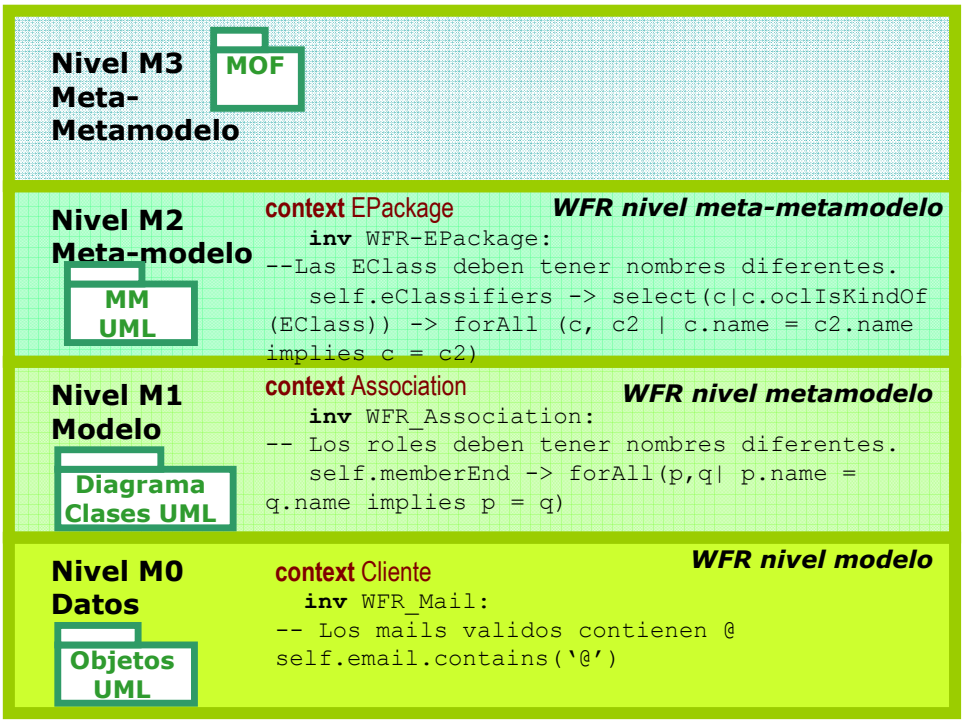
- Invariantes
- Pre y Post condiciones

Con estas restricciones se pueden definir reglas de buena formación en los distintos niveles.

Arquitectura 4 capas de modelado
OCL - Reglas de buena formación



Arquitectura 4 capas de modelado
OCL - Reglas de buena formación



Sintaxis Abstracta - Implementación

- Provee herramientas de modelado y generación de código.
- Utiliza una implementación de MOF llamada Ecore como meta-metamodelo.
- A partir del modelo definido, crea automáticamente el código que lo implementa.
- El código generado es claro y eficiente y puede ser modificado por el usuario y sus cambios se mantienen.

Sintaxis Abstracta - Implementación

- Crea para el modelo dado:
  - La interfaz e implementación de cada clase.
  - Mecanismos de notificaciones – Observer
  - Factory para instanciar.
  - Package para acceder a los metadatos del modelo.
  - Mecanismo de persistencia a XMI.
- Genera un editor del modelo en forma de árbol.
- Genera adaptadores para editar propiedades de instancias.
- Genera esqueletos para casos de prueba.

Sintaxis Abstracta - Implementación

- El plugin de OCL permite agregar restricciones y operaciones al modelo Ecore definido.
- Las mismas se incluyen vía anotaciones.
- A partir de ellas, genera código que incluirá la evaluación de las mismas usando el evaluador de OCL.

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

Sintaxis Concreta

- La sintaxis concreta define como se muestran los elementos del modelo al usuario.
- Puede ser en forma de diagrama o texto.

```
collaboration role Guionista;
collaboration role Lider;
tool TextEditor
    references "Herramienta de tex
shared object Guion;
session RedactarGuion
    roles ( Guionista -> Lider;);
workspace SalaReunion;
```

Sintaxis Concreta

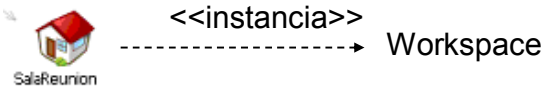
- Definir la sintaxis concreta sirve para:
  - ✓Interpretarla y asegurar su validez.
    Análisis léxico y sintáctico
  - ✓Instanciar la sintaxis abstracta.
- Rol crucial en el diseño del lenguaje, ya que lo lleva a los sentidos del usuario.

Sintaxis Concreta - Modelo

Su descripción consiste de:

•Un alfabeto con símbolos básicos

•Reglas de reconocimiento (scanning y parsing)

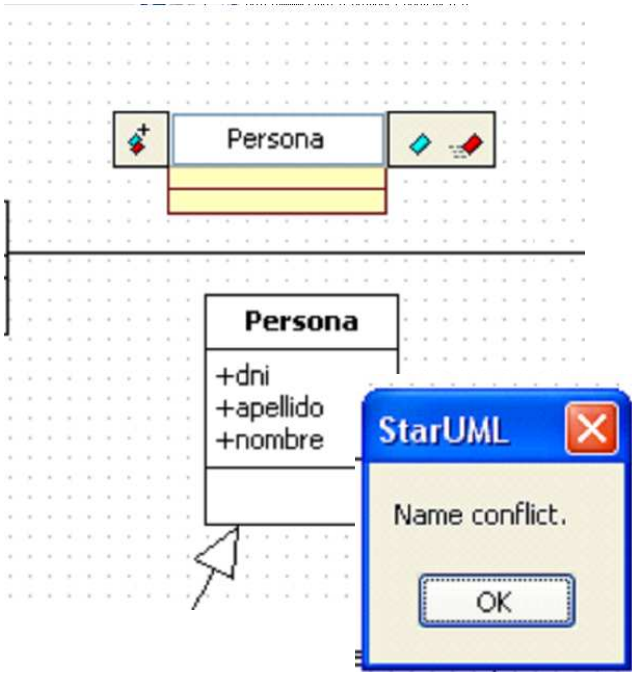


•Reglas de vínculos entre elementos (binding)

•Reglas de abstracción

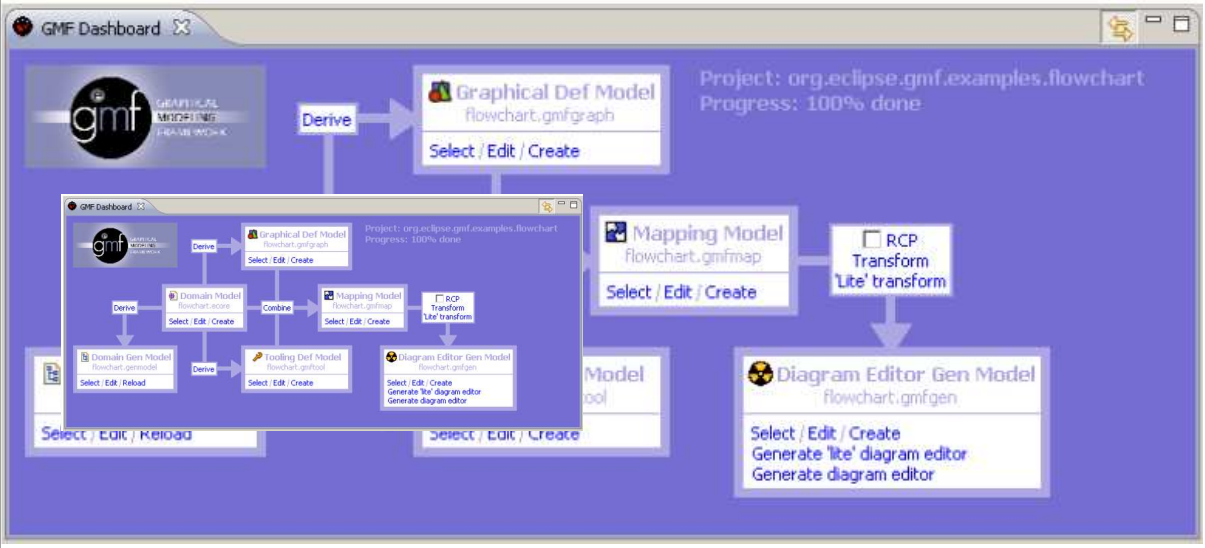
Sintaxis Concreta

- Dos tipos de editores
  - Libres: Completa libertad para crear el programa.
  - Estructurados: Solamente permiten crear estructuras válidas.



Plugins para la sintaxis concreta

Sintaxis Concreta - Gráfica



Sintaxis Concreta - Textual

- Genera componentes para editar y mostrar instancias del modelo, a partir de la especificación de la sintaxis del lenguaje.
- El editor generado incluye colores para los distintos tokens y sugerencias de texto.

- ATLAS del INRIA
- Permite definir una sintaxis textual para un metamodelo KM3
- Parsear y mostrar el modelo.
- En incubación.

- Parte del proyecto TMF de Eclipse.
- Se especifica la gramática del lenguaje y desde allí, el modelo Ecore.
- Disponible desde hace muy poco.

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

Semántica

- Describe el significado en términos de comportamiento, propiedades estáticas o la traducción a otro lenguaje.

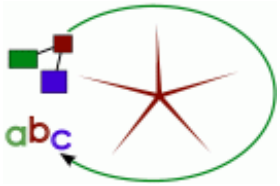
- Se puede describir de las siguientes maneras:
  - Por traducción: a otro lenguaje con semántica precisa.
  - Operacional: como se interpretan los conceptos.
  - Extensional: por extensión de semántica existente.
  - Denotacional: por medio de expresiones matemáticas.

Semántica - Implementación

JET

Java Emitter Template

- Java Emitter Template (JET): subproyecto de EMF que permite traducir modelos a texto.
- La salida se genera a partir de plantillas similares a JSP.
- Usado por el plugin de OCL.



•MOFScript: lenguaje de transformación que permite llevar cualquier modelo MOF a texto.

•Utiliza OCL para navegar entre los elementos del modelo de entrada.

•Más intuitivo.

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

DSL - Sistemas Colaborativos

Sistemas colaborativos: surgen de la unión de computadoras, con el objetivo de promover el trabajo en grupo.

También se ve como grupos de personas involucradas en un trabajo común, realizando tareas simultáneas.

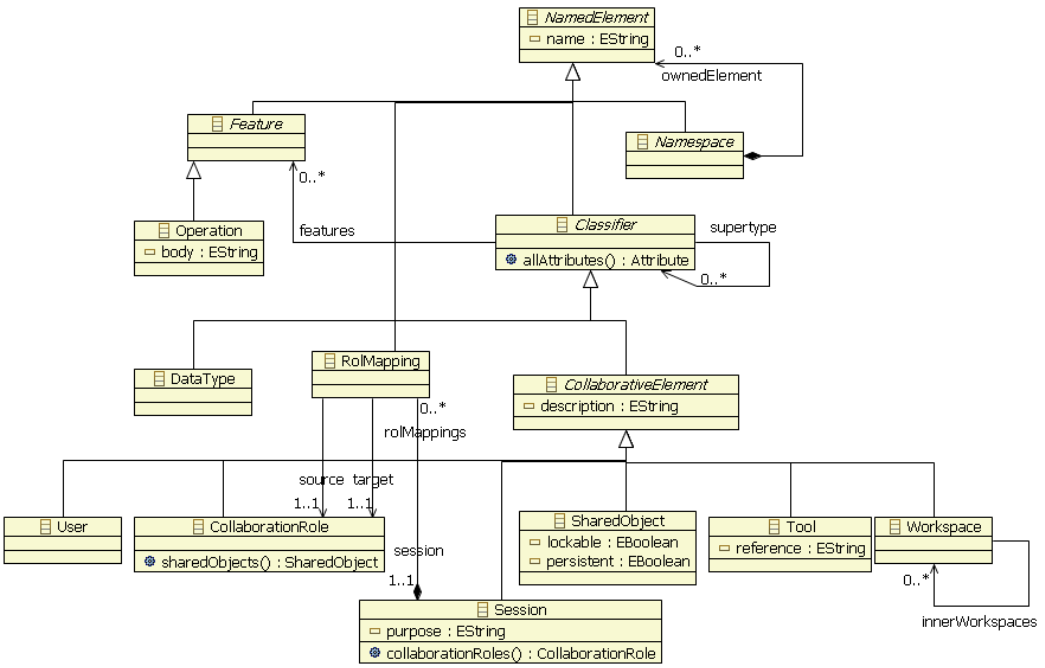
Sistemas colaborativos – Sintaxis Abstracta

Identificación de conceptos:

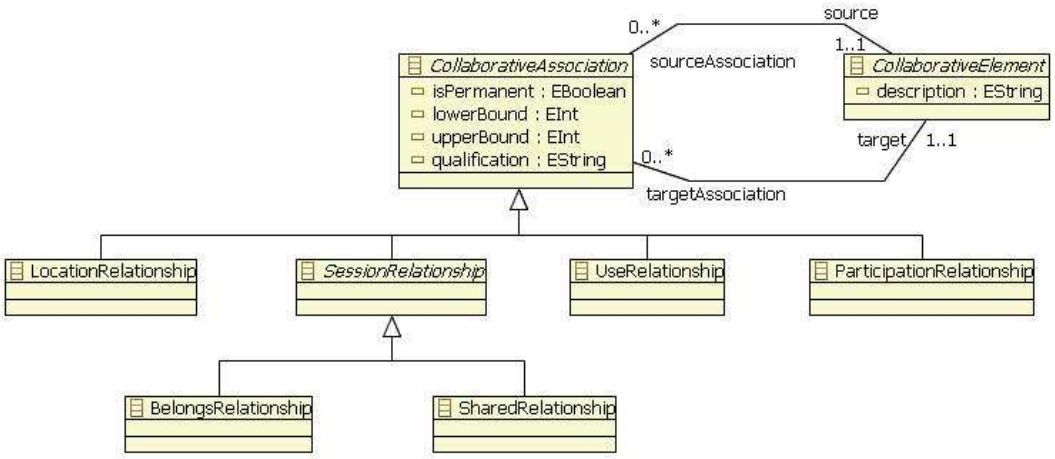
Entidades
- Usuario
- Rol
- Objetos compartidos
- Espacio de trabajo
- Sesión
- Herramientas colaborativas

Relaciones
- seUbicaEn
- perteneceA
- seComparteCon
- usa
- participaEn

Sintaxis abstracta - Entidades



Sintaxis abstracta – Relaciones



Sintaxis abstracta – Reglas OCL

- Los elementos conectados por medio de la relación sessionRelationship son instancias de espacios de trabajo y sesiones

- Una useRelationship se usa entre roles o sesiones, y las herramientas u objetos compartidos

```
inv validUseRelationship:
(self.source.oclIsKindOf(CollaborationRole) or
   self.source.oclIsKindOf(Session)) and
( self.target.oclIsKindOf(Tool) or
   self.target.oclIsKindOf(SharedObject) )
```

Temario

- Introducción
- Objetivo del trabajo
- Lenguajes Específicos de Dominio
  - Definición
  - Sintaxis Abstracta
  - Sintaxis Concreta
  - Semántica
- Construcción de un DSL
- Demo
- Conclusiones
  - Trabajos Futuros

Conclusiones - DSL

- La construcción de un DSL requiere un profundo conocimiento sobre el dominio y experiencia en la construcción de DSL.
- Libera al desarrollador de la tarea de escribir código. Pero se tiene que justificar el costo de su desarrollo.
- Permite mantener actualizado los modelos. Y sirven como documentación para los expertos del dominio.

Conclusiones - Herramientas

- Eclipse provee buen soporte para el desarrollo de DSLs.
- En la implementación, encontramos como aspectos positivos:
  - Implementar DLS sin escribir código.
  - Permite modificar el generado para adecuarlo a las necesidades.
  - Hay documentación disponible.
  - Se obtienen editores ricos, de alta calidad.

Conclusiones - Herramientas

- Aspectos negativos:
  - Problemas cuando evolucionan los metamodelos.
  - La variedad de herramientas dificulta la elección de la correcta.

Trabajo a futuro

- Se proponen los siguientes:
  - Herramienta integradora que vincule los distintos plugins.
  - Profundizar con el DSL para ambientes colaborativos.
  - Implementar una editor para figuras compuestas a usar en GMF.
  - Investigar como solucionar el problema de desincronización ante la evolución de los metamodelos.

Preguntas