UNIVERSIDAD AUTÓNOMA DE MADRID

Escuela Politécnica Superior

Un framework para la generación automática de ejercicios mediante técnicas de mutación

Pablo Gómez Abajo

Tutores: Esther Guerra y Juan de Lara

http://www.miso.es

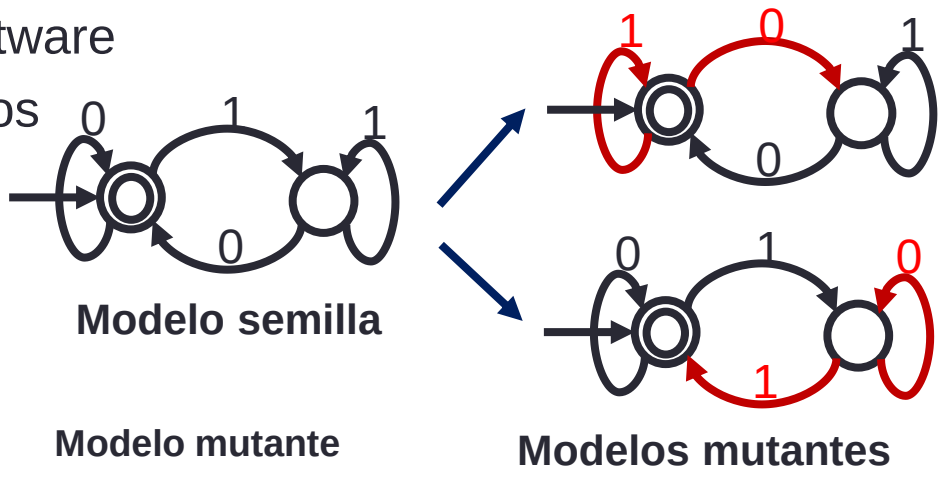
modelling & software engineering research group

Universidad Autónoma de Madrid

Trabajo Fin de Máster, Junio 2016

¿Qué es la mutación de modelos?

- La mutación de un modelo es la variación de un modelo semilla al que se aplica uno o más operadores de mutación
- La mutación de modelos tiene múltiples aplicaciones:
  - Pruebas de transformación de modelos
  - Pruebas de software basadas en modelos
  - Pruebas de líneas de producto software
  - Generación automática de ejercicios
  - Algoritmos genéticos
  - …



Objetivos

- Diseño e implementación de:
  - Un lenguaje de mutación independiente del dominio: **Wodel**
  - Una aplicación de este lenguaje a la generación automática de ejercicios: **Wodel-Edu**

- Motivación:
  - No hay lenguajes generales para la mutación de modelos
  - No hay frameworks para generar ejercicios de forma automática independientes del dominio

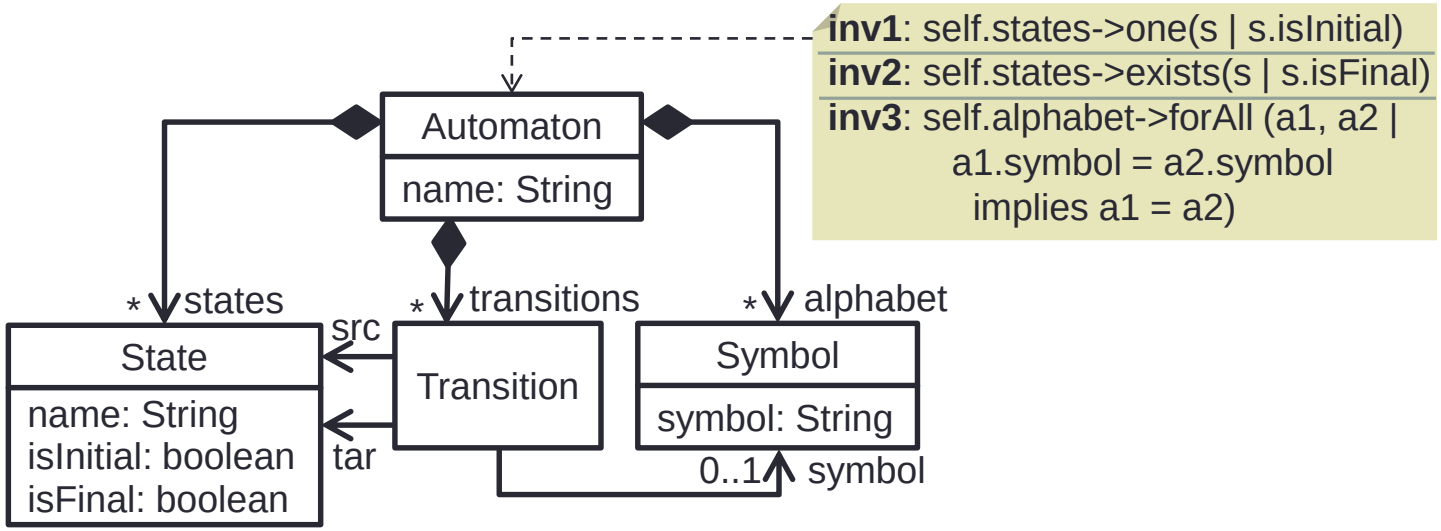
- Wodel-Edu sirve además como prueba de concepto de Wodel

Índice

I. Conceptos y técnicas de la Ingeniería Dirigida por Modelos

Modelos y meta-modelos

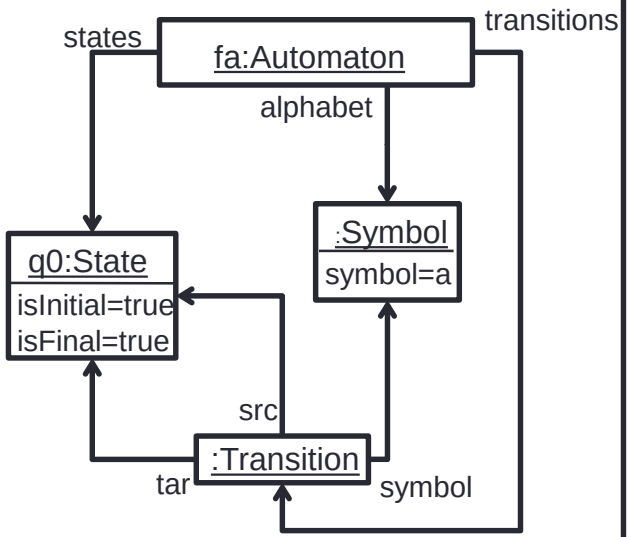
- Un modelo es la descripción de un sistema utilizando un lenguaje
- Un meta-modelo es un modelo que describe un lenguaje
- Los meta-modelos se definen mediante diagramas de clase o diagramas entidad-relación
- Restricciones OCL adicionales



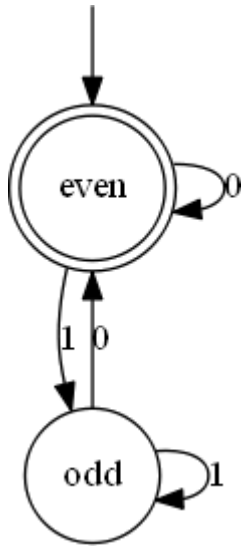
Sintaxis abstracta y sintaxis concreta

- Un meta-modelo define la **sintaxis abstracta** del lenguaje
- La **sintaxis concreta** incluye información sobre cómo visualizar y representar los conceptos de la sintaxis abstracta

Modelo de autómata finito



Sintaxis concreta gráfica



Sintaxis concreta textual (hecho con Xtext)

```
Automaton acceptEvenBinary {
    states {
        even(isInitial, isFinal),
        odd
    }
    transitions {
        src even tar even symbol '0',
        src even tar odd symbol '1',
        src odd tar even symbol '0',
        src odd tar odd symbol '1'
    }
    alphabet {
        '0',
        '1'
    }
}
```

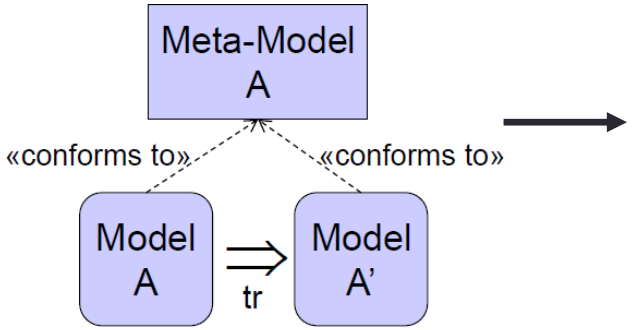
Lenguajes de dominio específico (DSLs)

- Un DSL ofrece los conceptos para un dominio de aplicación
- Gráfico, o textual
- Primitivas de alto nivel
- Orientado a expertos en el dominio
- Traducido automáticamente a lenguajes como Java, C, etc.
- Wodel es un DSL para aplicar mutaciones en modelos

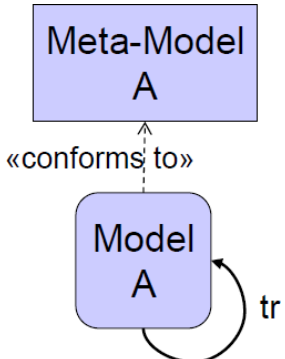
Transformación de modelos

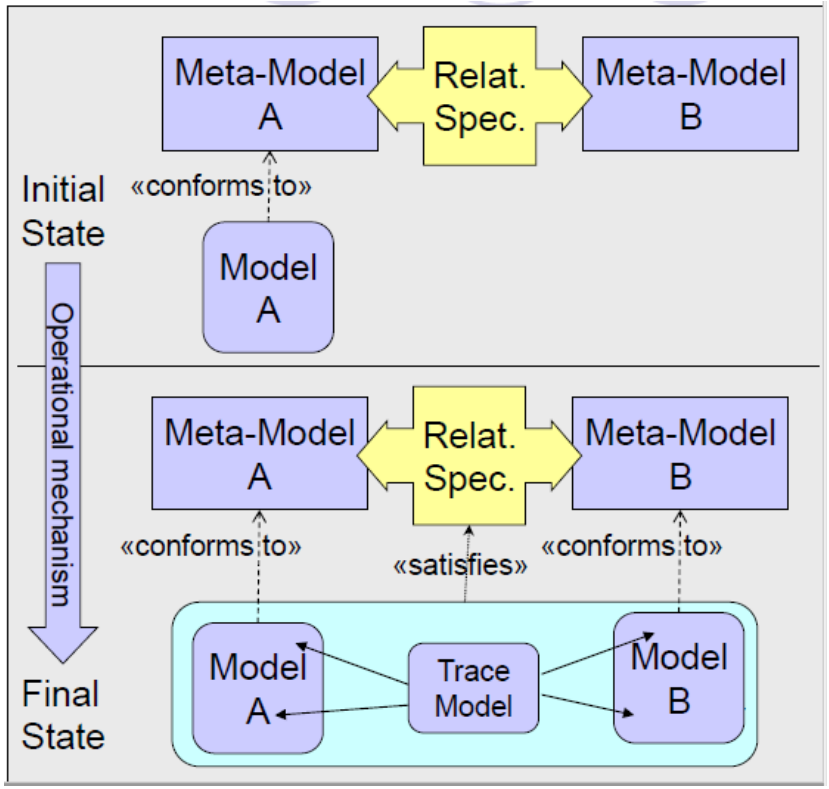
- Necesario para la manipulación de modelos

- Out-place



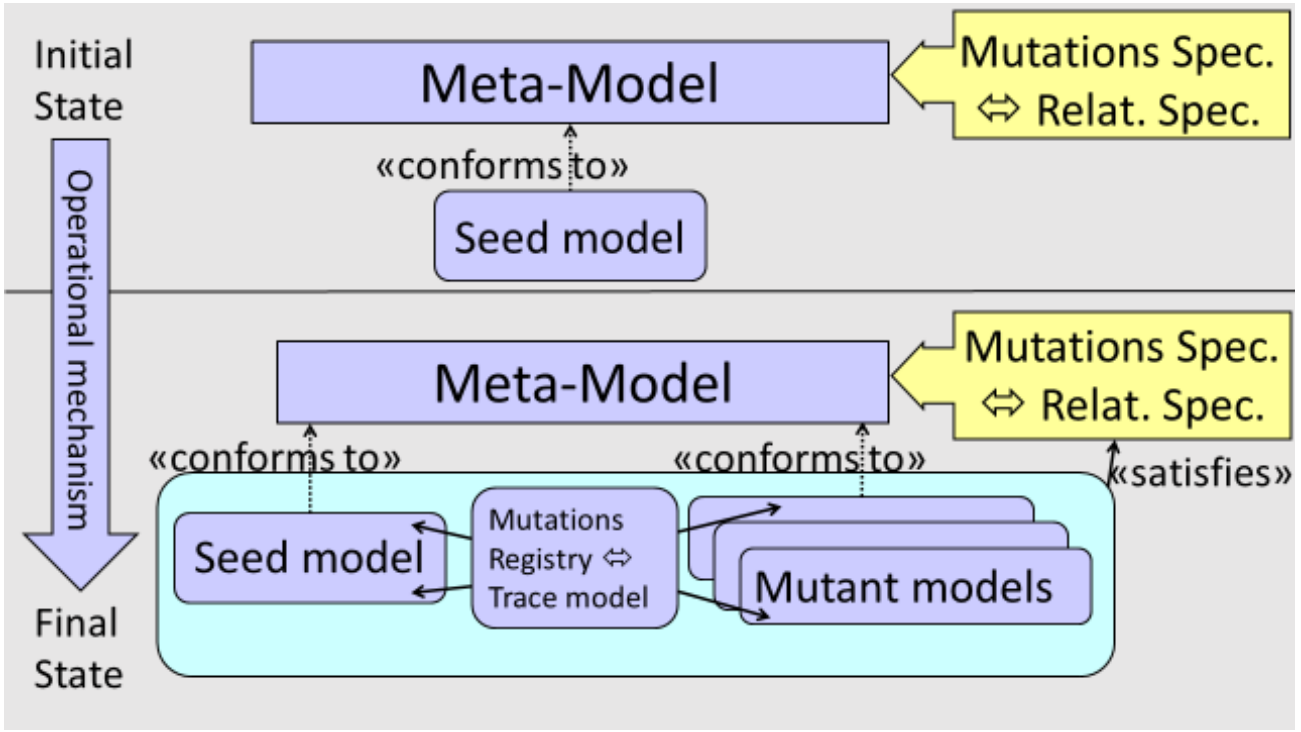
- In-place





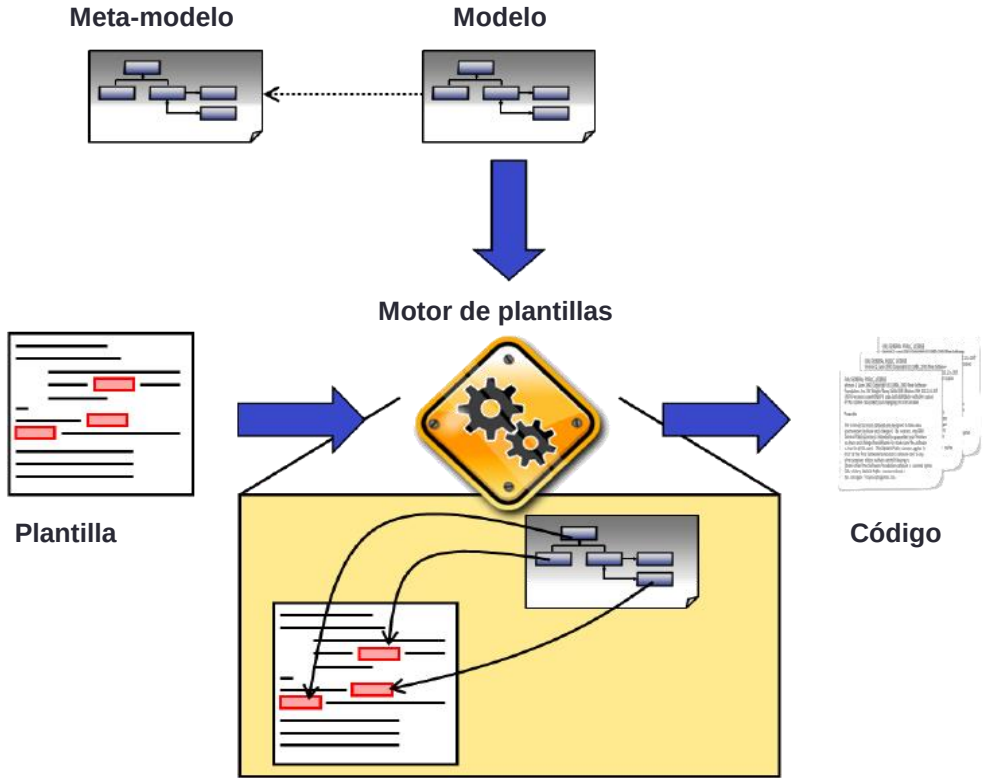
Mutación de modelos

- Tipo especial de transformación de modelos
- Los modelos de entrada son los modelos semilla
- Los modelos de salida son los mutantes
- Los modelos semilla y los mutantes son conformes al mismo meta-modelo



Generación de código

- Producción de código a partir de un modelo de un nivel más alto con el objetivo de crear una aplicación que funcione
- Código Java de los programas Wodel
- Código HTML y JavaScript de la aplicación web creada con Wodel-Edu

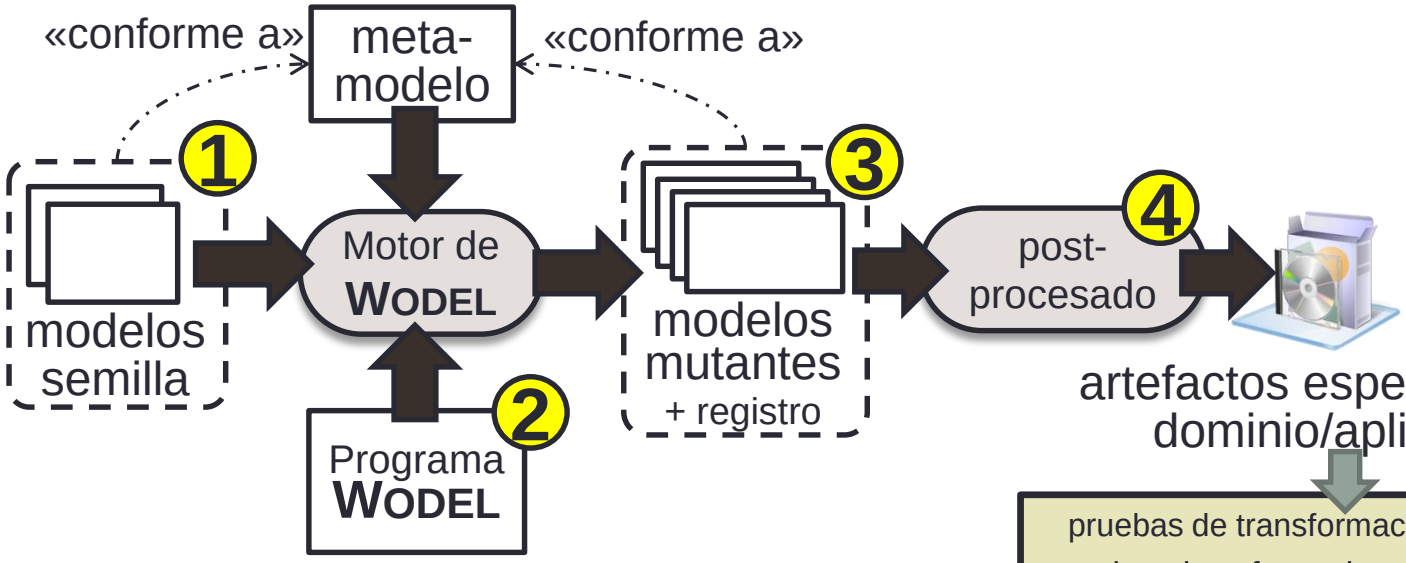


II. Wodel: un lenguaje de dominio específico para mutación de modelos

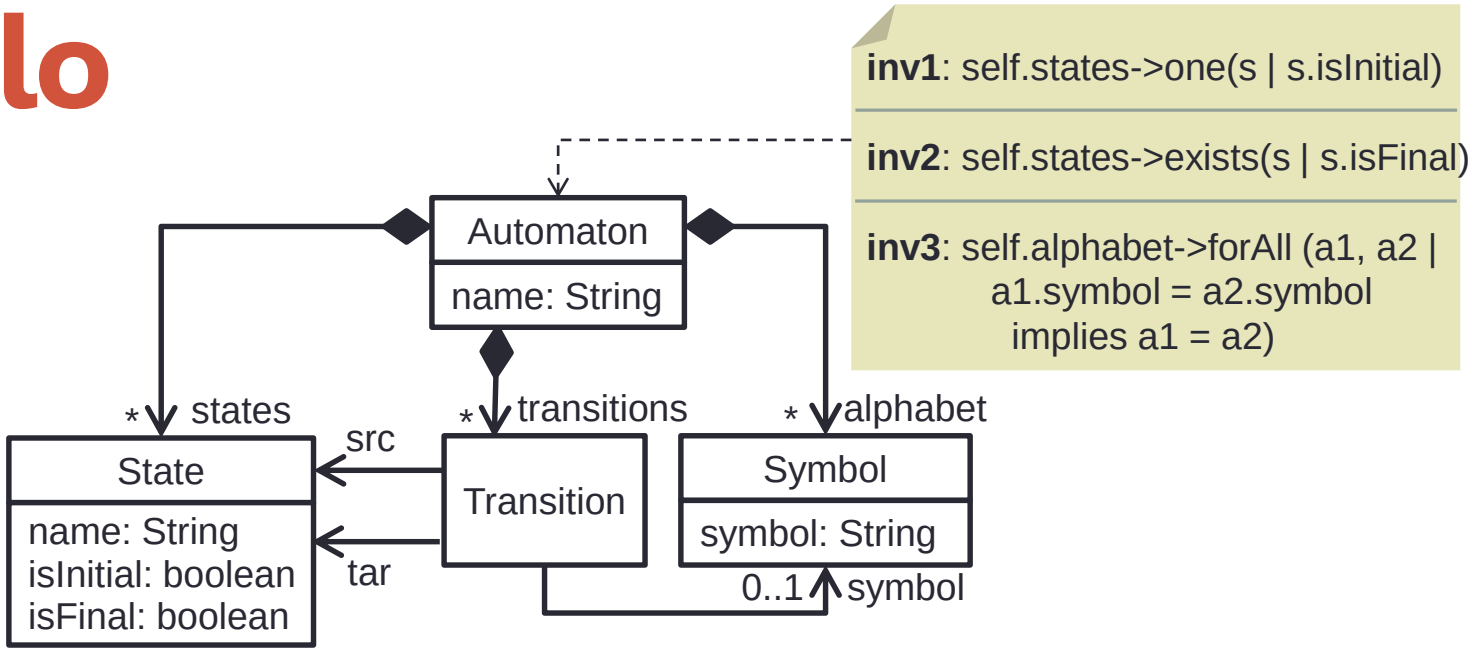
Motivación

- Los frameworks existentes para mutación de modelos son:
  - Específicos para un lenguaje (p.ej., fórmulas lógicas)
  - Específicos para un dominio (p.ej., pruebas)
  - Los operadores de mutación están codificados a mano
- Se propone el DSL **Wodel** para mutación de modelos, con:
  - Primitivas de mutación de alto nivel
  - Independencia del lenguaje y del dominio destino
  - Compilado a código Java
  - Extensible mediante post-procesadores

Esquema


«conforme a»
meta-modelo
«conforme a»
1
modelos semilla
Motor de WODEL
2
Programa WODEL
3
modelos mutantes
+ registro
4
post-procesado
artefactos específicos del dominio/aplicación
pruebas de transformación de modelos
pruebas de software basadas en modelos
pruebas de líneas de producto software
generación automática de ejercicios
algoritmos genéticos
...

Ejemplo

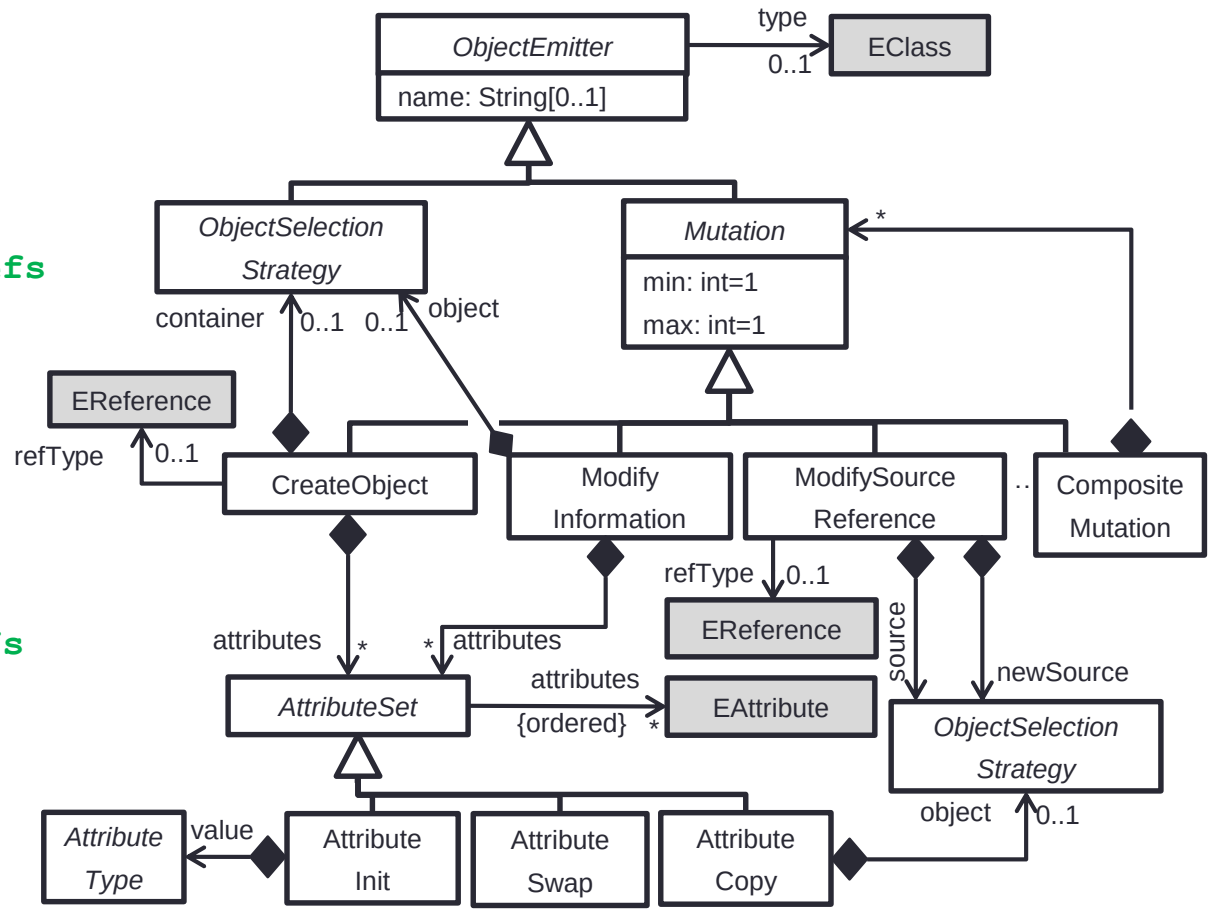



```
generate 3 mutants in "out/" from "evenBinary.fa"
metamodel "http://fa.com"

with commands {
  s0 = modify one State where {isFinal = true} with {reverse(isFinal)}
  s1 = create State with {isFinal = true}
  t0 = create Transition with {src = s0, tar = s1, symbol = one Symbol}
}
```

Operadores de mutación

```
// creación de objetos/refs
create State
create reference tar in
  one Transition
// modificación de objetos/refs
modify one State
  with {isFinal = true}
modify source src from
  one Transition
modify target tar from
  one Transition
// eliminación de objetos/refs
remove one State
remove reference tar in
  one Transition
// mutaciones compuestas
[
  s0 = create State
  modify s0 with {name = 's0'
]
```



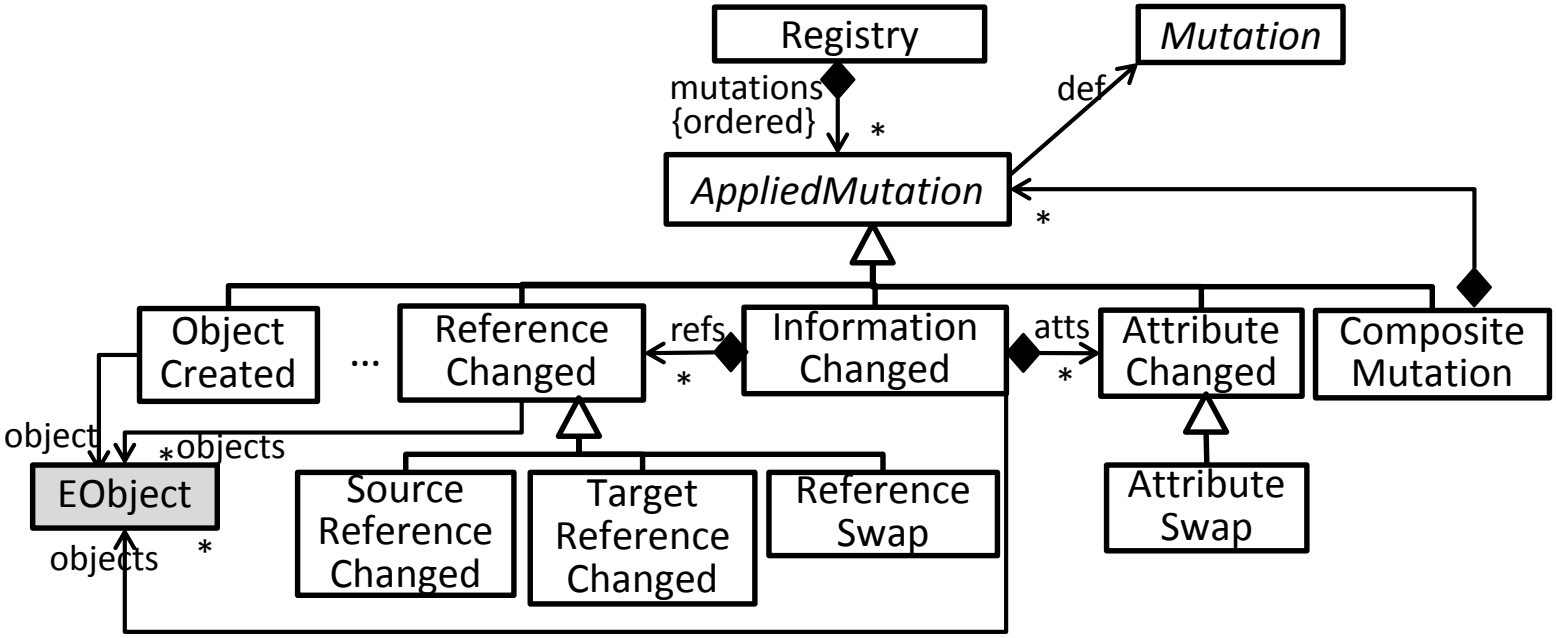
Asigna automáticamente los elementos creados a los contenedores
Elimina las referencias que quedan sueltas al borrar elementos
Comprueba las cardinalidades y restricciones OCL del meta-modelo
Control de mutantes duplicados
Compilado automático a código Java

Bloques y restricciones OCL

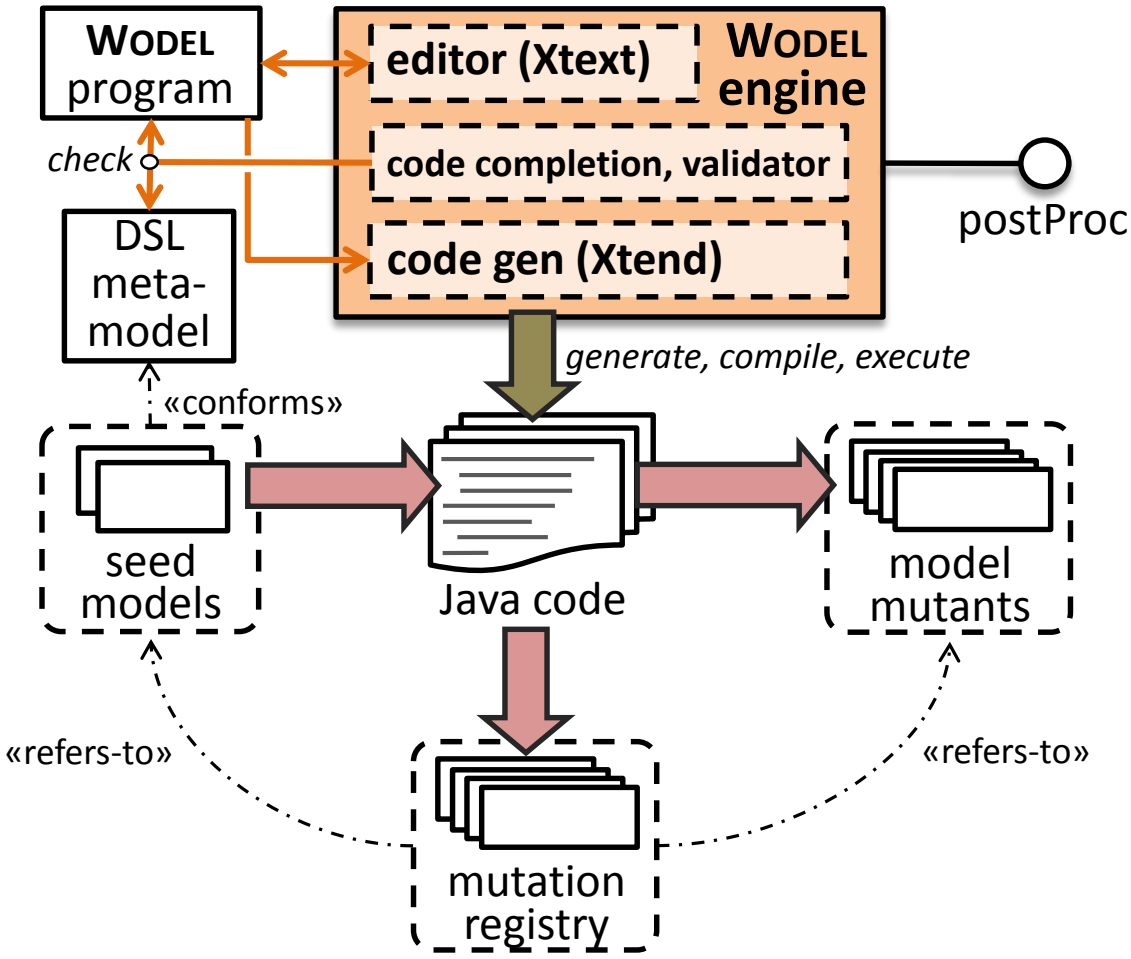
- Wodel da soporte para bloques de mutaciones:
  - Generación de mutantes por etapas
  - Un bloque puede tomar como modelos semilla los mutantes generados en bloques declarados previamente
  - Jerarquía de carpetas para identificación de los mutantes
  - Control de mutantes repetidos con la directiva repeat=no
  - Útil para los ejercicios de selección de opciones de texto en Wodel-Edu
- Restricciones OCL en el código Wodel:
  - Se aplican sobre los modelos mutantes generados, aunque no estén en el meta-modelo del dominio

Registro de mutaciones

- Opcional, se activa desde la página de preferencias
- Útil en la generación las opciones de texto en Wodel-Edu
- Referencias a modelos semilla y a modelos mutantes
- Opción de compactar el registro (mutaciones irrelevantes)

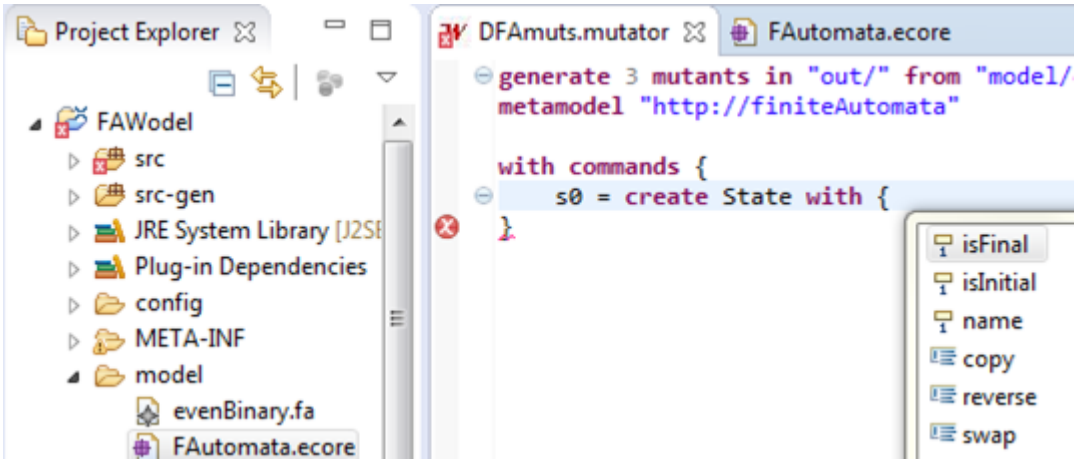


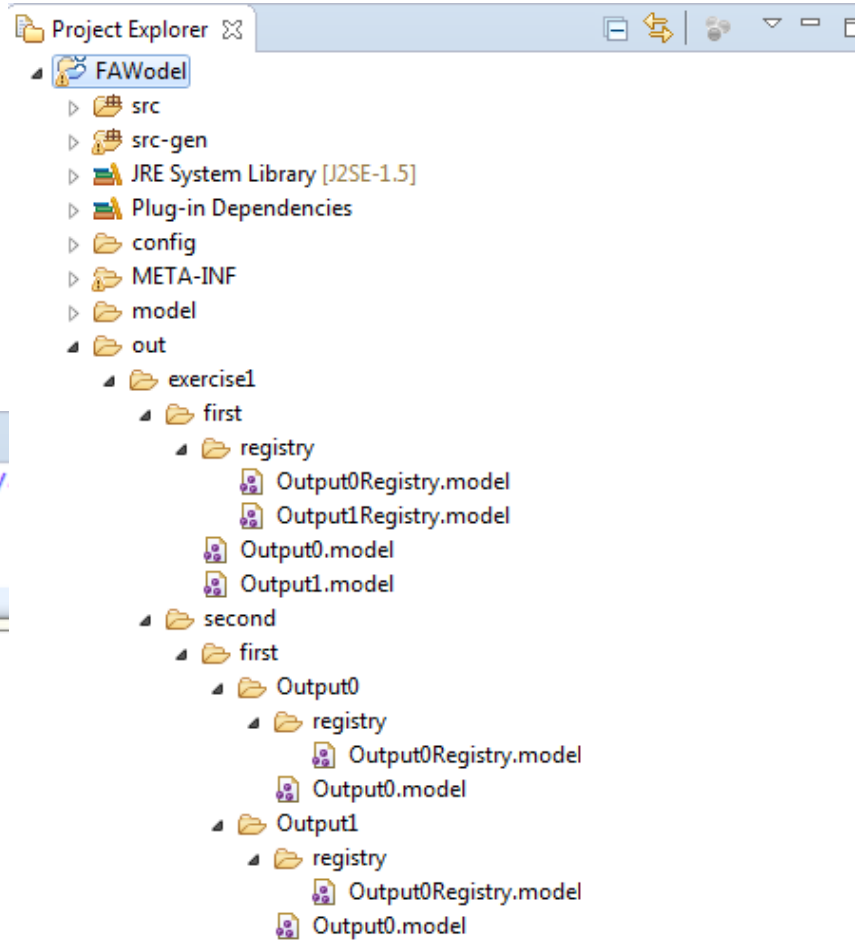
Arquitectura de la herramienta



IDE de la herramienta

- Completado de código, validación, generación de código, ...





Comparación de Wodel con Java

```
create Transition with {symbol = one Symbol}
```

```
1....
2.// create transition
3.EClass transitionClass = (EClass)epackage.getEClassifier("Transition");
4.EObject transition = EcoreUtil.create(transitionClass);
5.
6.// search object automaton in model
7.EObject automaton = null;
8.for (TreeIterator<EObject> it = seed.getAllContents(); it.hasNext();) {
9.   automaton = it.next();
10.  if (automaton.eClass().getName().equals("Automaton")) {
11.      // add transition to automaton
12.      EStructuralFeature feature =
13.         automaton.eClass().getEStructuralFeature("transitions");
14.      ((List<EObject>)automaton.eGet(feature)).add(transition);
15.      // set random state as source of the transition
16.      feature = automaton.eClass().getEStructuralFeature("states");
17.      List<EObject> states = (List<EObject>)automaton.eGet(feature);
18.      EObject randomState = states.get(rand.nextInt(states.size()));
19.      feature = transitionClass.getEStructuralFeature("src");
20.      transition.eSet(feature, randomState);
21....
```

Facilidad de integración con otros programas Java

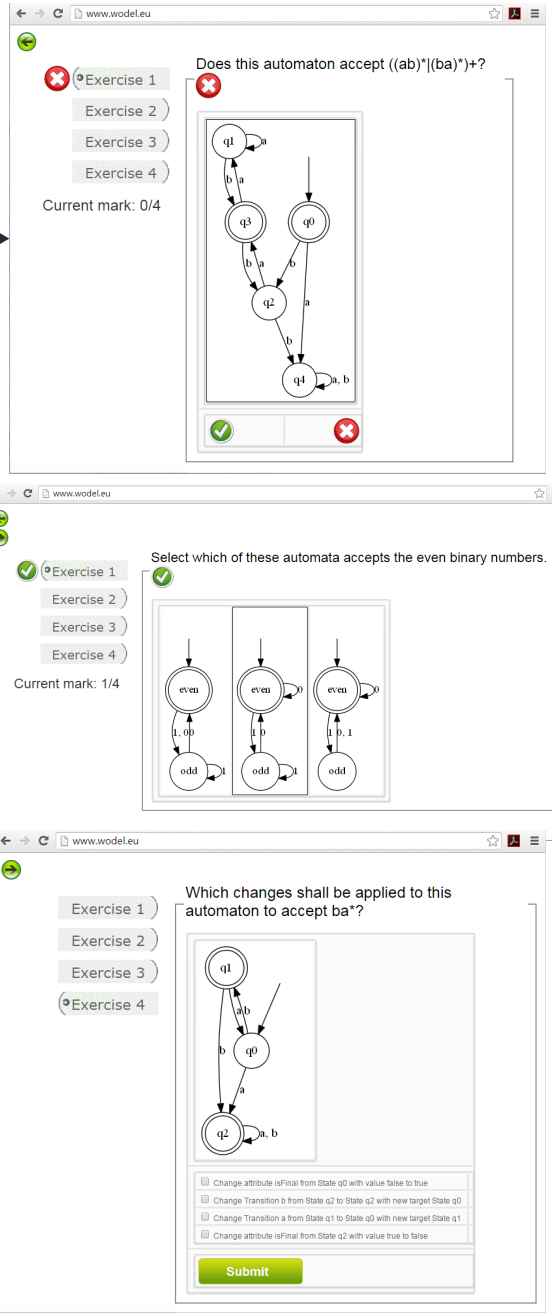
III. Wodel-Edu: un framework para la generación automática de ejercicios mediante técnicas de mutación

Motivación

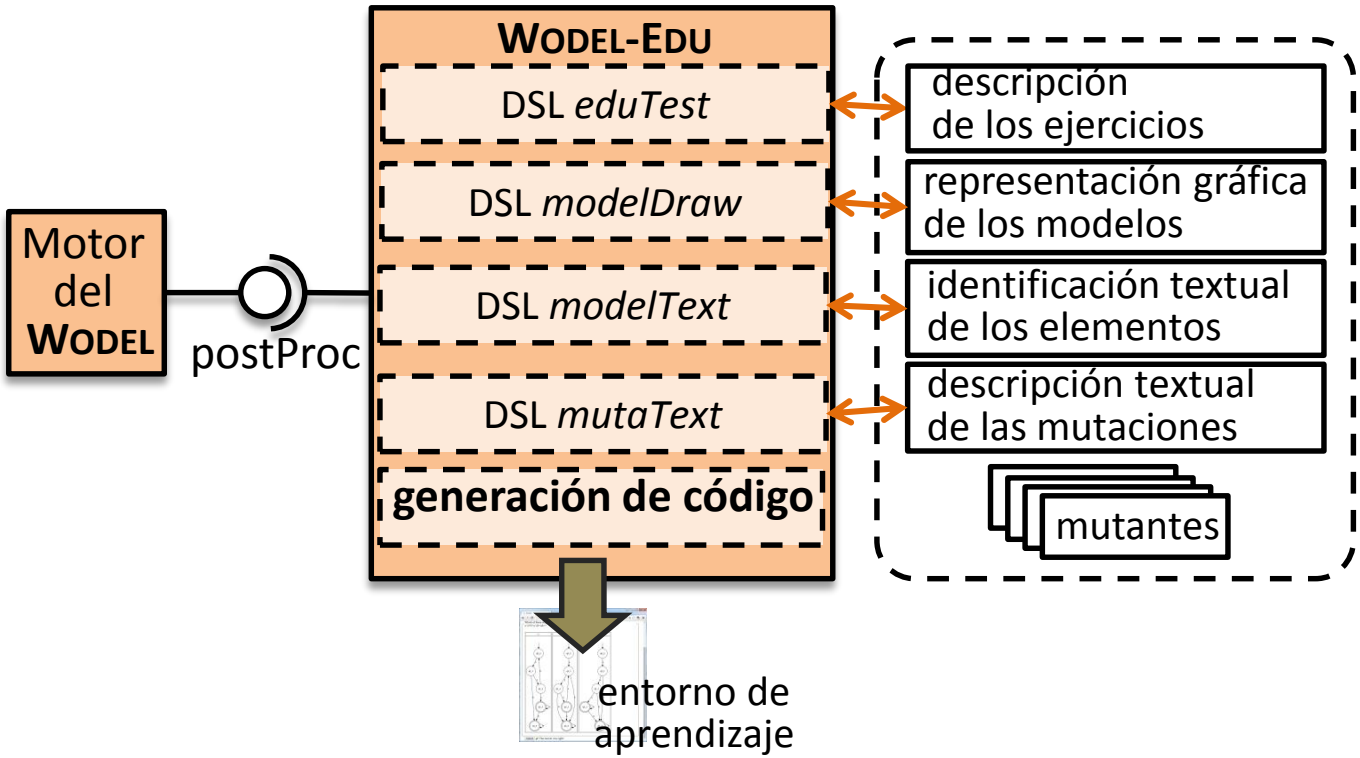
- Se propone el entorno **Wodel-Edu** – una extensión a Wodel – para la generación automática de ejercicios que es independiente del dominio

- Aplicación web con tres formatos de ejercicios:
  - Respuesta alternativa
  - Selección de un diagrama entre varios
  - Selección de opciones de texto

- Necesidad de:
  - Representación gráfica de los modelos
  - Generación de las opciones de texto
  - Generación de código HTML+JavaScript

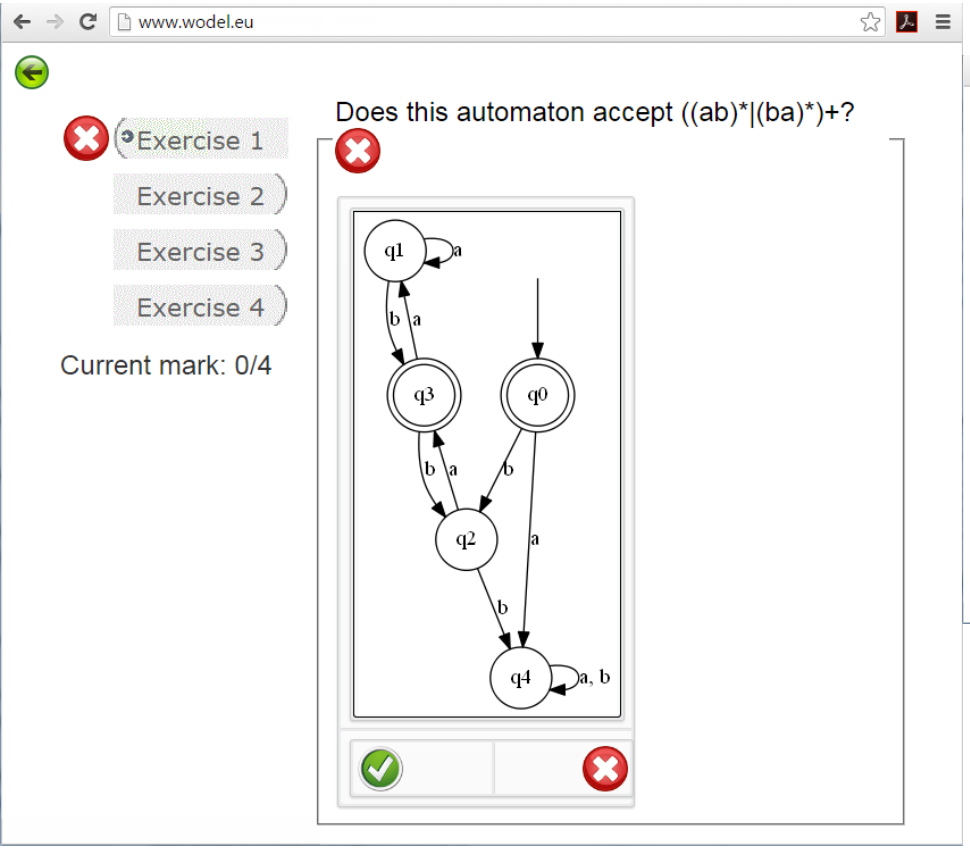


Arquitectura

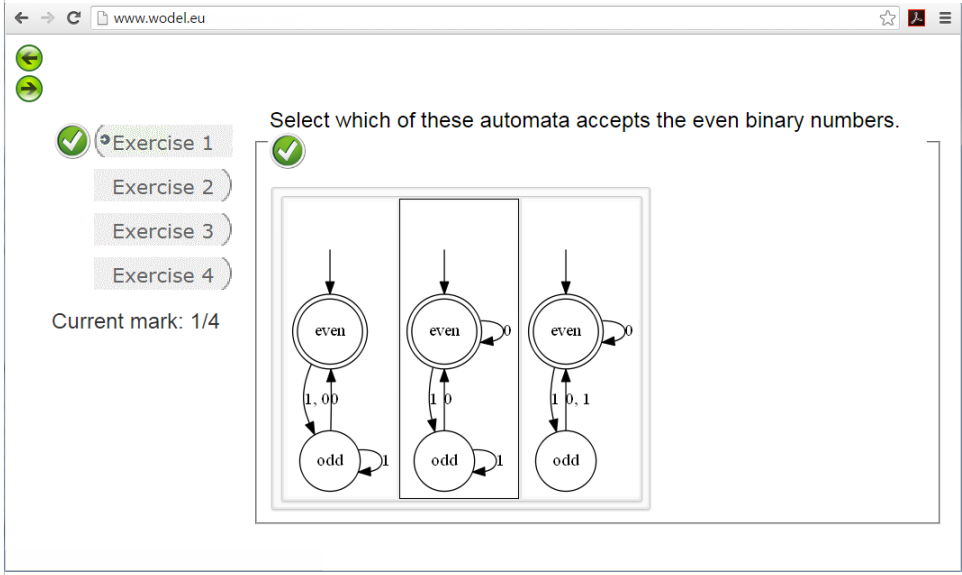


Formatos de ejercicios I y II

- Respuesta alternativa



- Selección de un diagrama entre varios

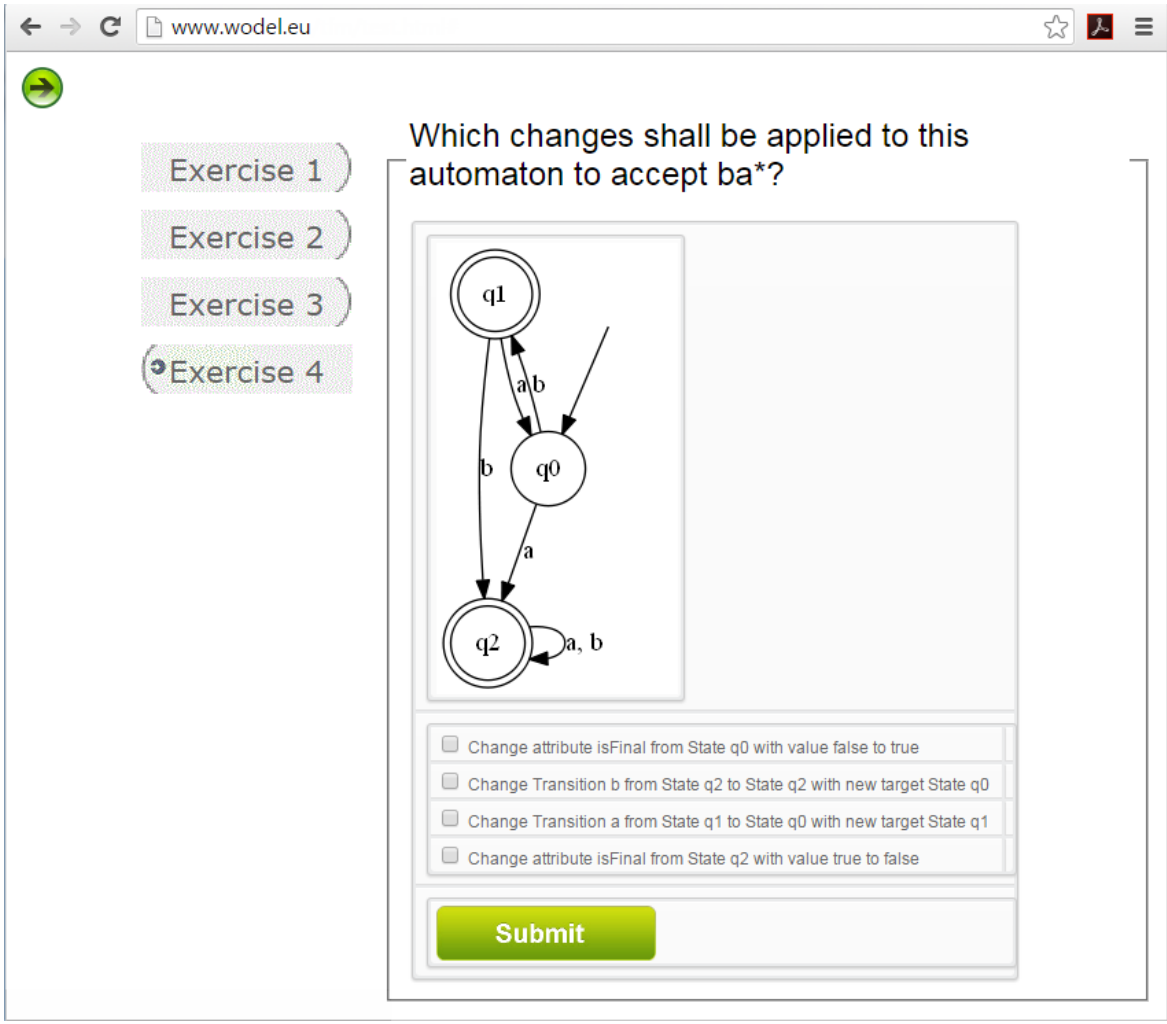


Formato de ejercicios III

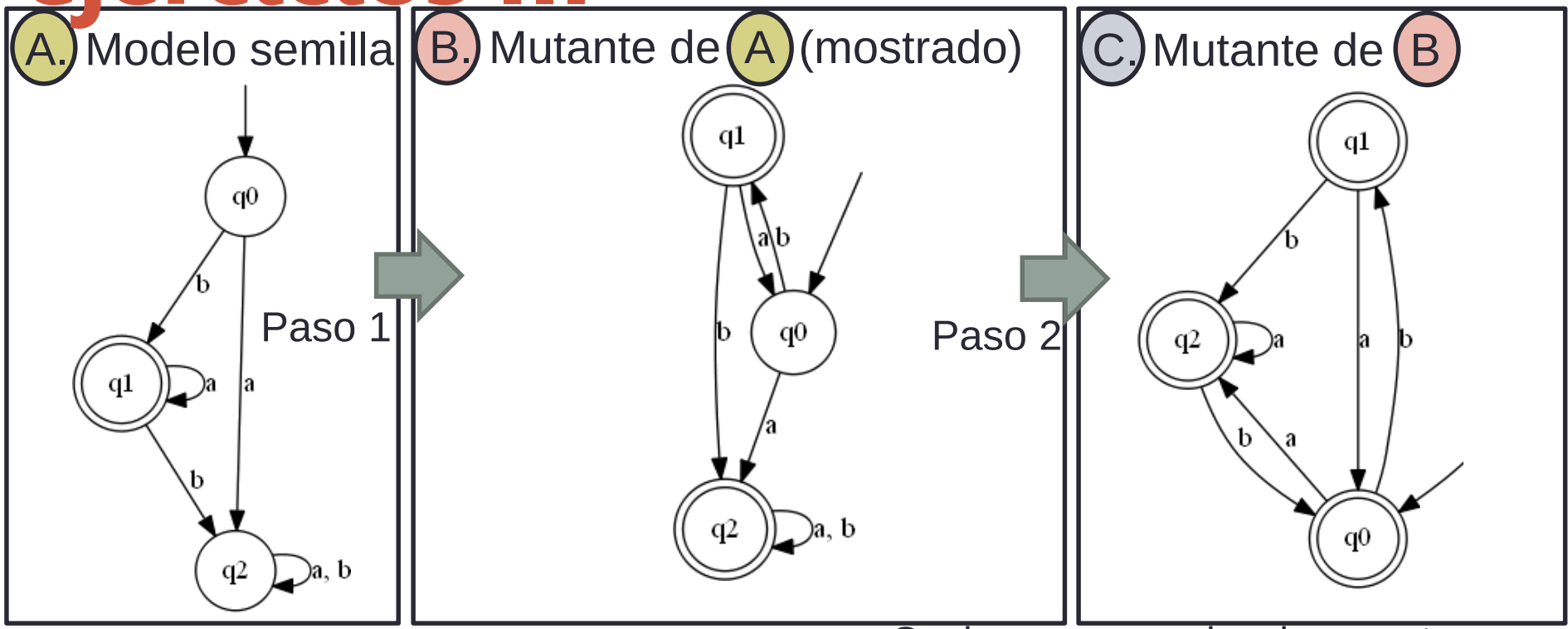
- Selección de opciones de texto para corregir el diagrama

- Las opciones de texto (correctas e incorrectas) se generan utilizando el registro de mutaciones

- Los DSLs modelText y mutaText se pueden utilizar para la configuración del texto de estas opciones



Descripción del formato de ejercicios III



- Opciones generadas: correctas

1. Change attribute isFinal from State q2 with value true to false
2. Change Transition a from State q1 to State q0 with new target State q1

- Opciones generadas: incorrectas

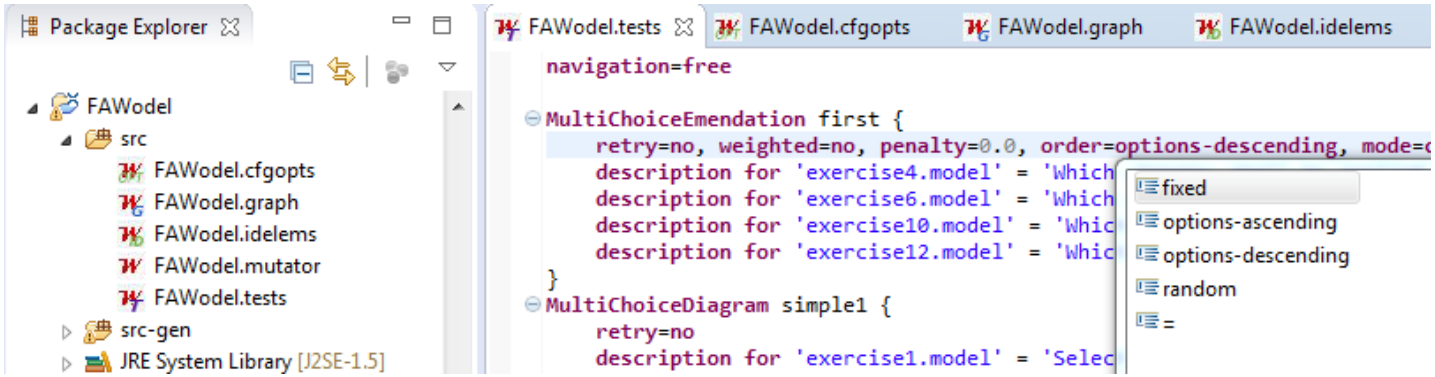
1. Change attribute isFinal from State q0 with value false to true
2. Change Transition b from State q2 to State q2 with new target State q0

Mutaciones de los pasos 1 y 2:

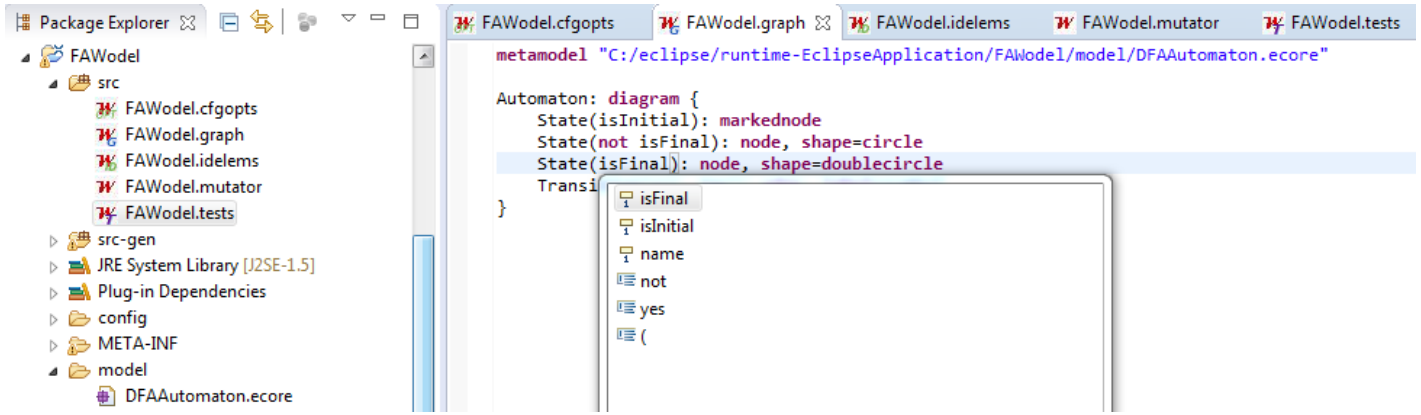
```
modify target tar from one Transition to other State
modify one State with {reverse(isFinal)}
```

IDE de Wodel-Edu I

- DSL eduTest

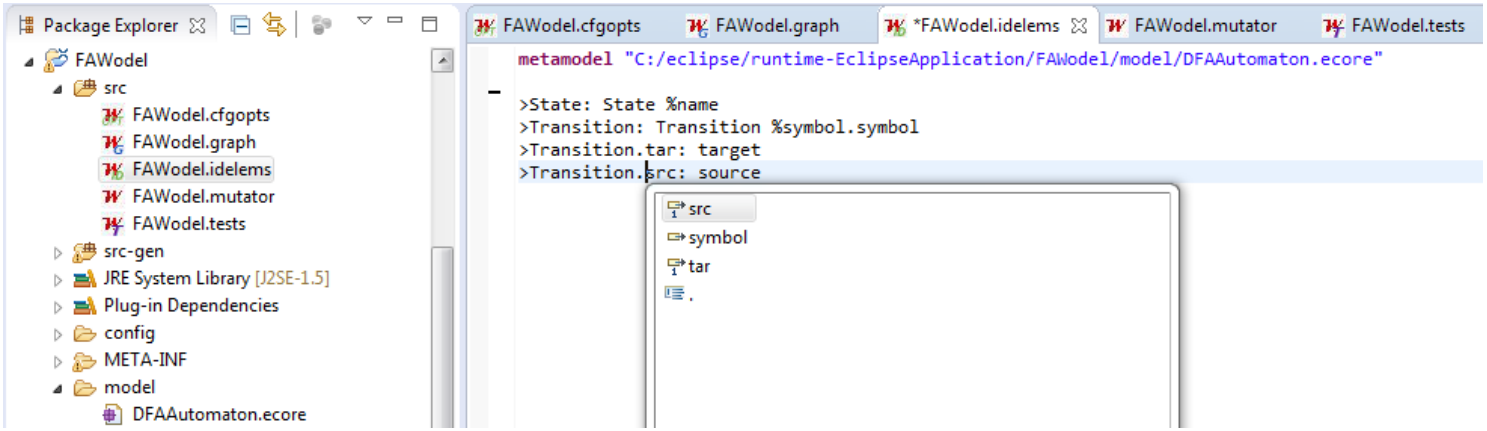


- DSL modelGraph

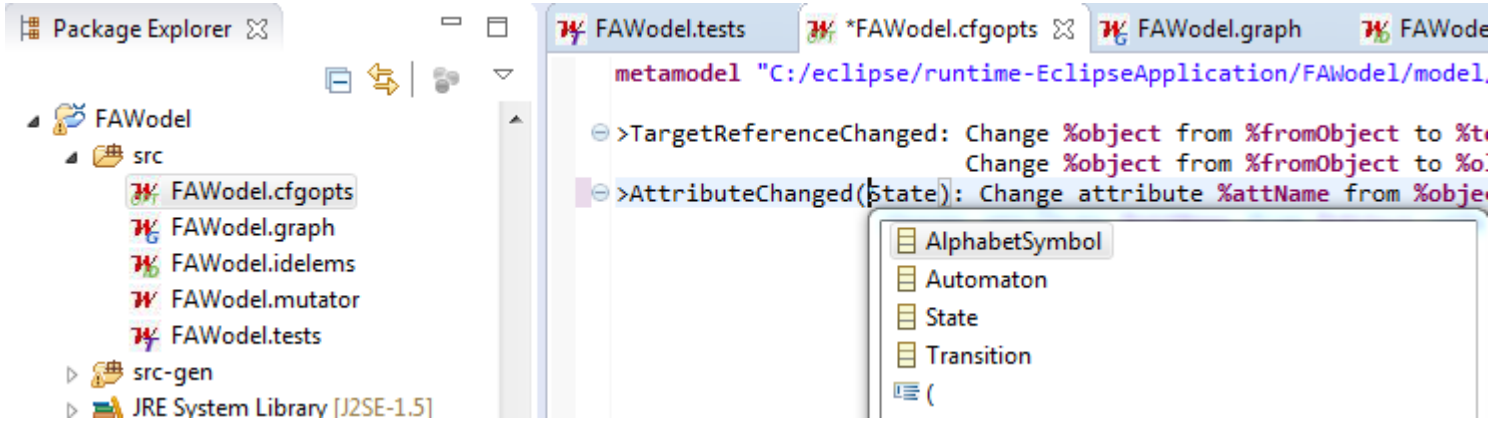


IDE de Wodel-Edu II

- DSL modelText



- DSL mutaText

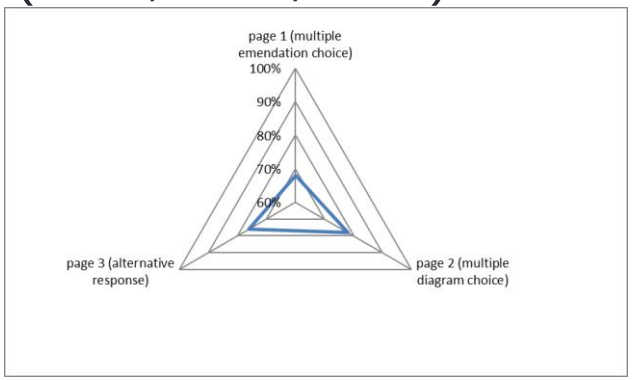


Descripción de la evaluación

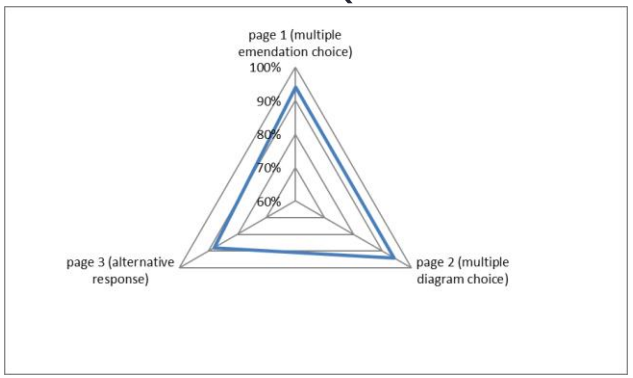
- Se genera una aplicación web de ejercicios de test con Wodel-Edu (http://www.wodel.eu):
  - Primera página: 4 ejercicios de opciones de texto
  - Segunda página: 4 ejercicios de seleccionar un diagrama entre varios
  - Tercera página: 4 ejercicios de respuesta alternativa
- Se miden tres dimensiones, además de la nota obtenida (opcional):
  - El ejercicio se entiende bien
  - La dificultad del ejercicio es adecuada
  - El ejercicio es útil para aprender autómatas
- Se consiguen 10 participantes (1 sin formación en autómatas, 8 hombres y 2 mujeres, entre 22 y 41 años, ...)

Resultados de la evaluación

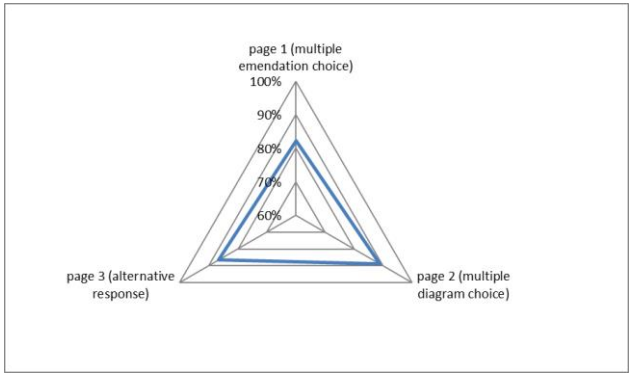
- Fácil de entender (68%, 78%, 76%)



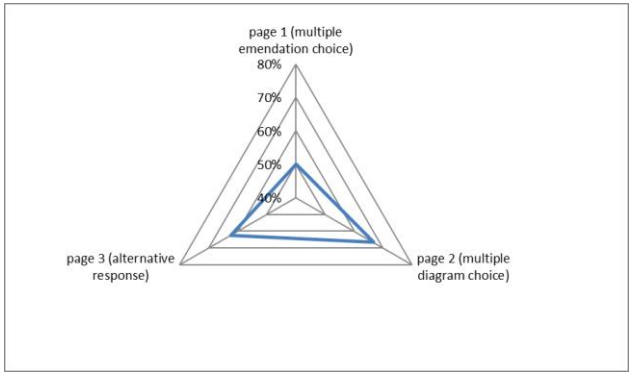
- Útil para aprender autómatas (94%, 94%, 88%)



- Nivel adecuado de dificultad (82%, 89%, 87%)



- Nota obtenida (50%, 67%, 63%)



Conclusiones de la evaluación

- El ejercicio de selección de opciones de texto es complicado de entender:
  - Agrupar con mode=radiobutton en el DSL eduTest
  - Mostrar el resultado de aplicar la opción sobre el diagrama
  - Hacer el ejercicio interactivo
- Dificultad percibida razonable
- Se perciben como muy útiles para aprender autómatas

- Cuestiones para evaluaciones posteriores
  - Incluir instrucciones o un tutorial
  - Influencia del orden de los ejercicios (mejor dificultad creciente)
  - Mezclar ejercicios generados de forma automática con otros hechos a mano
  - Evaluación desde el punto de vista del profesor

IV. Conclusiones y trabajo futuro

Conclusiones

- Wodel es un DSL para mutación de modelos:
  - Primitivas de mutación de alto nivel
  - Independiente del dominio
  - Da soporte a mutaciones compuestas
  - Da soporte a bloques de mutaciones: mutar mutantes generados previamente
  - Mutantes duplicados, validez de los mutantes
  - Compilado a Java
  - Extensible para diferentes aplicaciones

- Wodel-Edu: generación automática de ejercicios mediante técnicas de mutación
  - Independiente del dominio
  - Tres tipos diferentes de ejercicios
  - Generación de opciones de texto basadas en el registro de mutaciones
  - Resultados prometedores para la educación

Publicaciones

- Artículo ***'Wodel: a Domain-Specific Language for Model Mutation'*** presentado en el **31st ACM Symposium on Applied Computing (SAC'16)** que fue celebrado el pasado mes de abril en Pisa (Italia)

- Artículo ***'A Domain-Specific Language for Model Mutation and its Application to the Automated Generation of Exercises'*** para la revista **Computer Languages, Systems and Structures (Elsevier)** (en evaluación)

Trabajo futuro

- Ampliar Wodel con nuevas primitivas de mutación
- Desarrollar nuevos plugins para Wodel (para pruebas basadas en modelos, algoritmos genéticos…)
- Ampliar Wodel-Edu para generar entornos de aprendizaje más complejos (p. ej., con gamificación), ejercicios (p. ej., que sean interactivos), y diferentes plataformas (móviles o tablets)
- Hacer más experimentos con Wodel-Edu, contando con el punto de vista del profesor

Puedes descargarte el código
de este proyecto en GitHub:
http://gomezabajo.github.io/Wodel/
Una demo breve:
https://youtu.be/T9n3T0jGvzg
Gracias!!
Pablo.GomezA@uam.es