A DSL FOR MODEL MUTATION AND ITS APPLICATIONS TO DIFFERENT DOMAINS

Pablo Gómez-Abajo

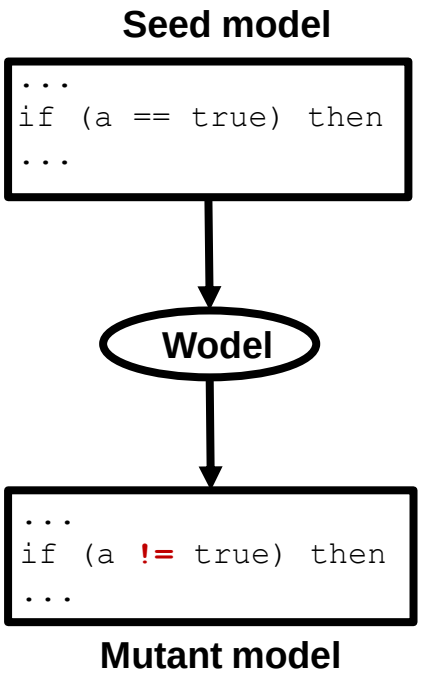
Modelling&Software Engineering Research Group
http://miso.es
Universidad Autónoma de Madrid (Spain)

UNIVERSIDAD AUTONOMA DE MADRID

Doctoral Symposium - MODELS'2016
Saint-Malo (France)

miso

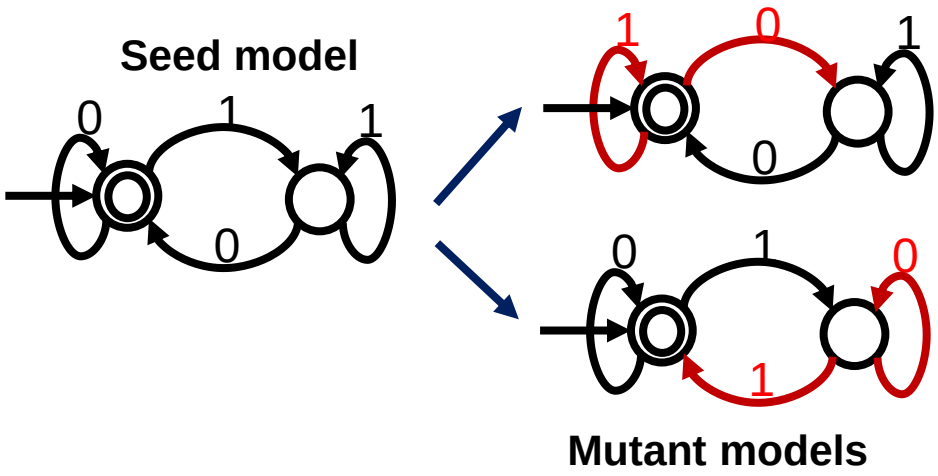
WHAT IS A MODEL MUTATION?



A model mutation is a variation of a seed model by the application of one or more mutation operators.

Model mutation has many applications:

- Model transformation testing
- Model-based software testing
- Software product lines testing
- Automated generation of exercises
- Search-based engineering
- ...



PROBLEM

The existing frameworks for model mutation are:

- Specific for a language (e.g., logic formulae)
- or a domain (e.g., testing)
- Mutation operators are usually manually encoded

There is a lack of proposals facilitating:

- The definition of mutator operators
- Applicable to arbitrary languages and applications

These would facilitate the creation of domain-specific mutation frameworks

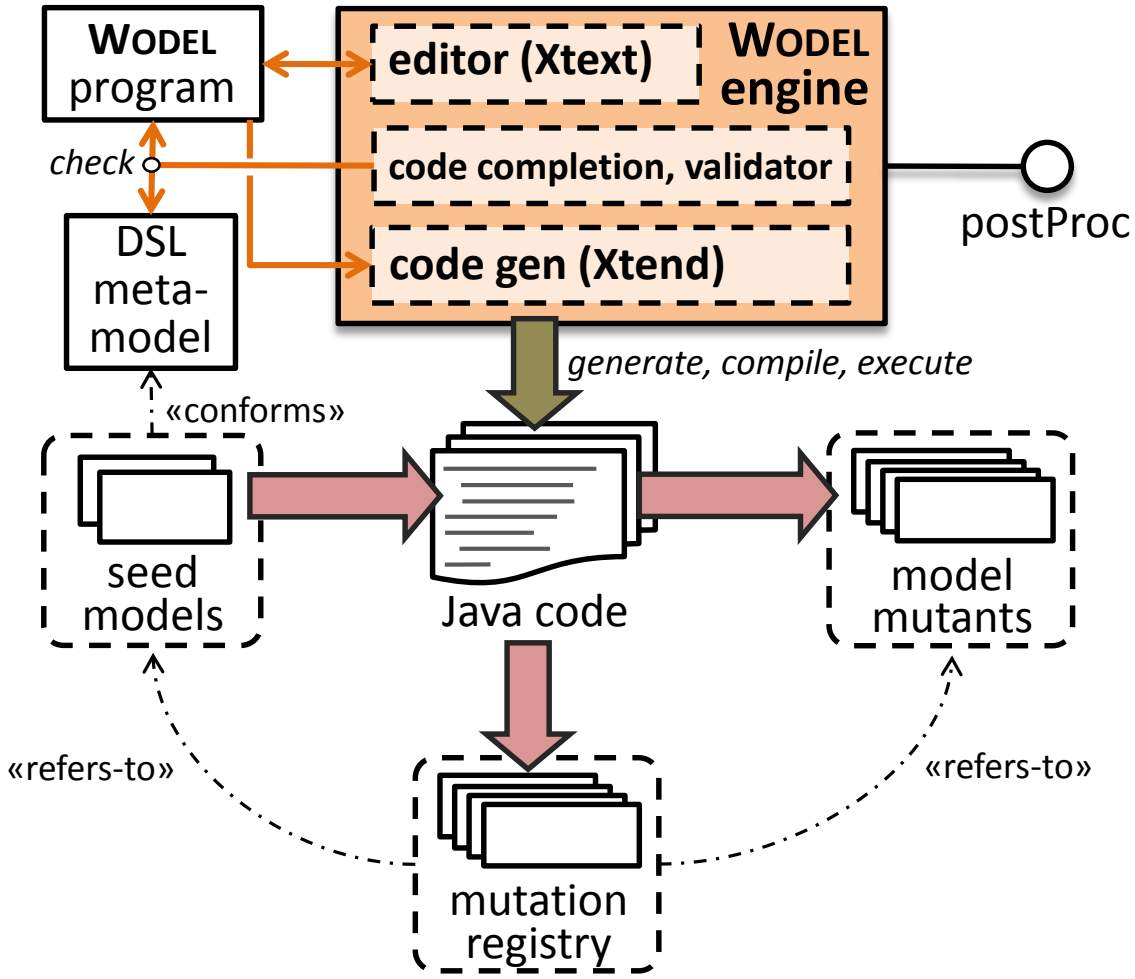
PROPOSED SOLUTION

We propose Wodel, a DSL to facilitate the specification and creation of model mutations in a meta-model independent way, with:

- High-level primitives (creation, deletion, modification, etc.)
- Different selection strategies (specifically, randomly, etc.)
- Independence of target language and domain
- Compiled to Java code
- Extensible through post-processors
- Mutation registry
- Blocks of mutations
- Support for OCL constraints inside Wodel code
- Execution policies

…

TOOL ARCHITECTURE



WODEL FEATURES

Support for cardinalities

- Apply mutations *n* times, or a number between *min* and *max*

Composite mutations

- Execute a set of mutations in one step

Correct/Incorrect mutant generation

- Mutants may satisfy seed meta-model, or not

Duplicate mutant identification

- Not only syntactically, but also semantically

Mutation primitives

- Creation, deletion, edge redirection, cloning, etc.

Model element selection strategies

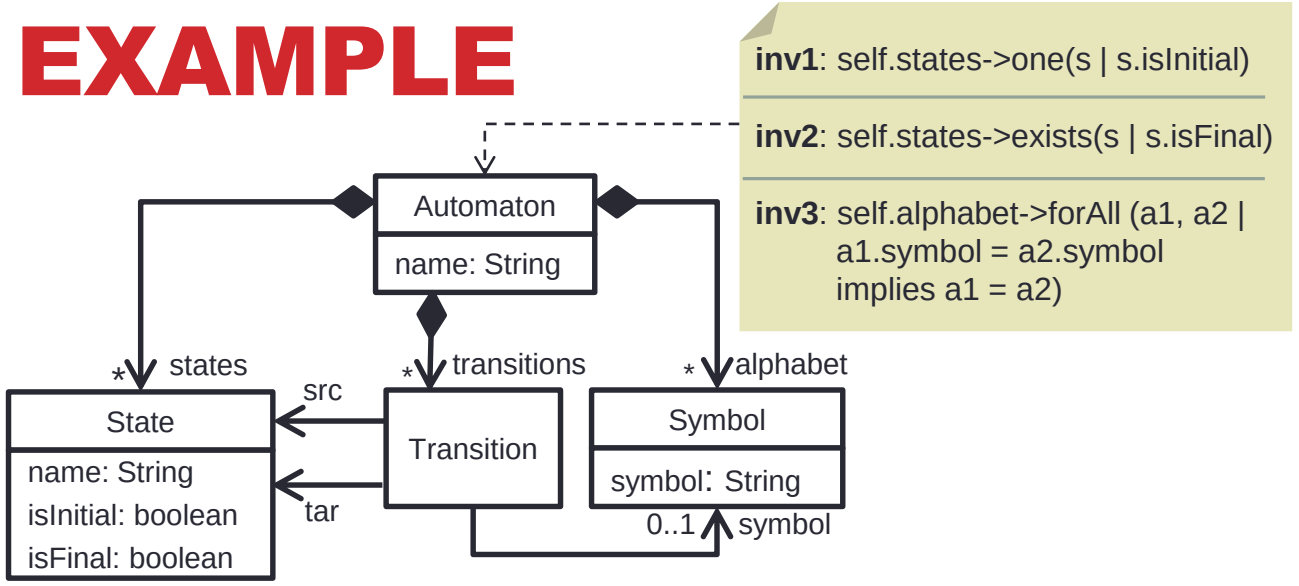
- Random selection, property-based selection

Execution policies

- Parallel, sequential, distributed

Libraries of reusable mutations for particular domains

WODEL: EXAMPLE




```
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}
}
```

APPLIED MUTATIONS REGISTRY

Optional, it is activated through the preferences page

References to seed models and mutant models

Optionally, the framework can reduce the registry (irrelevant mutations)

Repeatability

- It will be posible to repeat the mutation process

Verbalize applied mutations forward and backward

- Text options in the automated generation of exercises

BLOCKS AND OCL CONSTRAINTS

Wodel supports mutation blocks:

- Mutants generation by stages
- A block can take as seed models the mutants generated in previous blocks
- Folders hierarchy for mutants identification
- Duplicated mutants control with directive repeat=no

OCL Constraints in Wodel Code:

- Applied over the generated mutants, although they are not in the domain meta-model

EXPECTED CONTRIBUTIONS

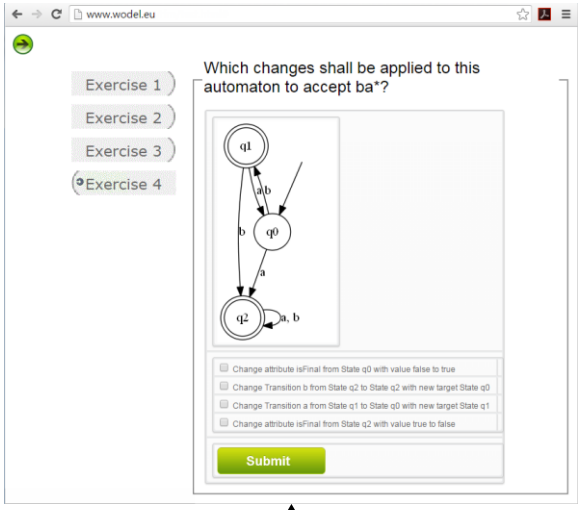
DSL Wodel will ease the creation of applications based on mutations by providing support for their:

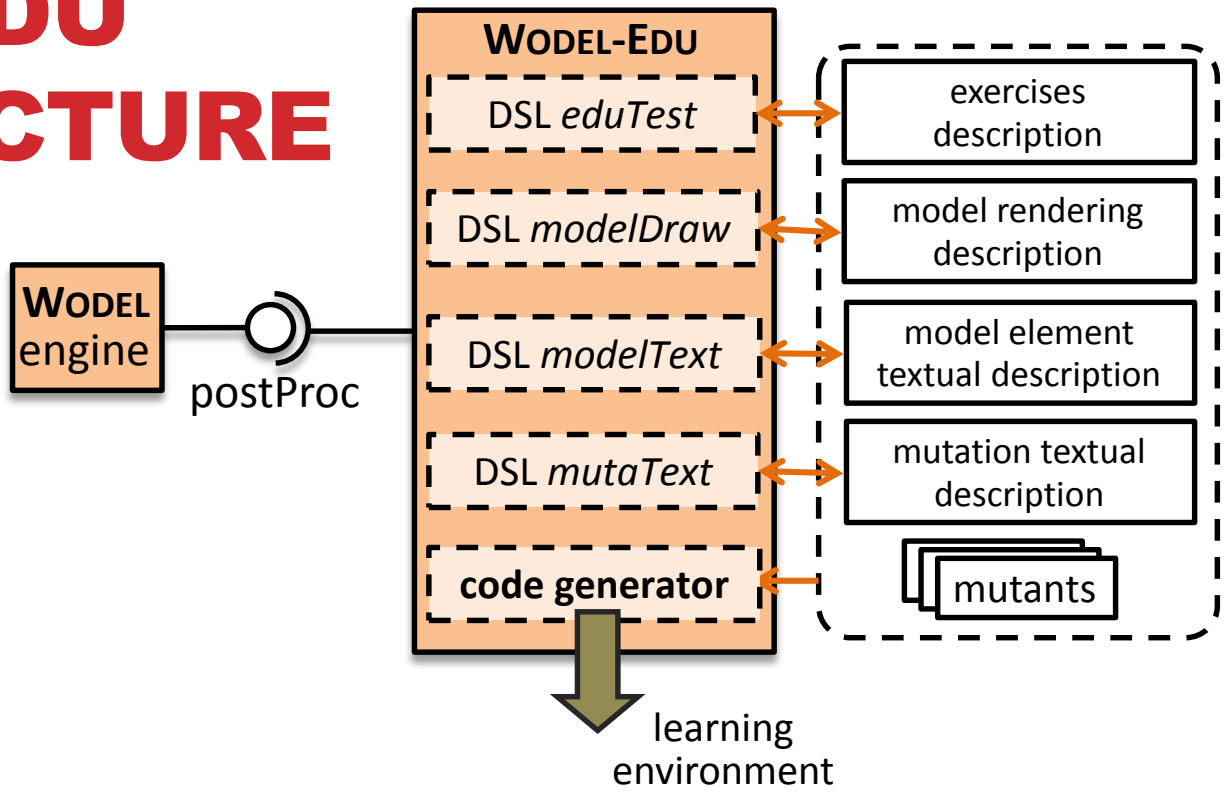
- Definition
- Execution
- Traceability

We will develop these three post-processing extensions:

- **Wodel-Edu:** Automated generation of exercises
- **Wodel-Unit:** Mutation-based software testing
- **Wodel-SB:** Search-and model-based software engineering

WODEL-EDU ARCHITECTURE



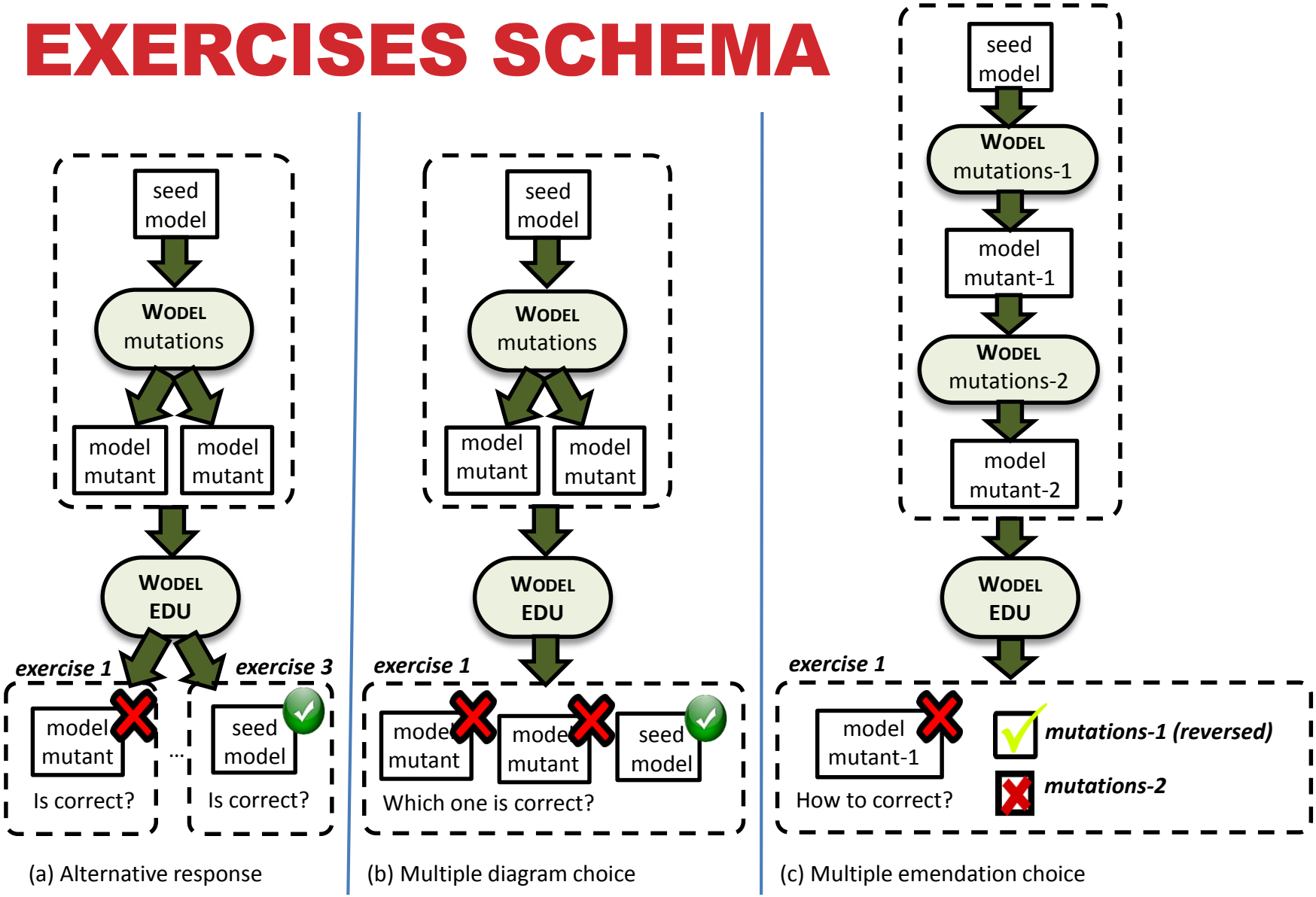


Wodel-Edu is a post-processing extension to Wodel for the automated generation of exercises

Wodel-Edu generates a web application (http://www.wodel.eu) with three kinds of test exercises:

- Alternative response
- Multiple diagram choice
- Multiple emendation choice

WODEL-EDU EXERCISES SCHEMA



EVALUATION AND VALIDATION

- **Evaluate the expressivity of Wodel coding in it the interesting mutations for different domains devised by us and found in the literature**
- **Test the Wodel-Edu generated exercises in real university courses (Automata Theory, Electronic Circuits, etc.)**
- **Use the software testing verification framework with real software projects, with the collaboration of the industry**
- **Use the approach to test ATL model transformations, complementing the previous work of our group**
- **Test the search-based engineering environment with the collaboration of some experts in this area within our department (http://aida.ii.uam.es/)**

CURRENT STATUS

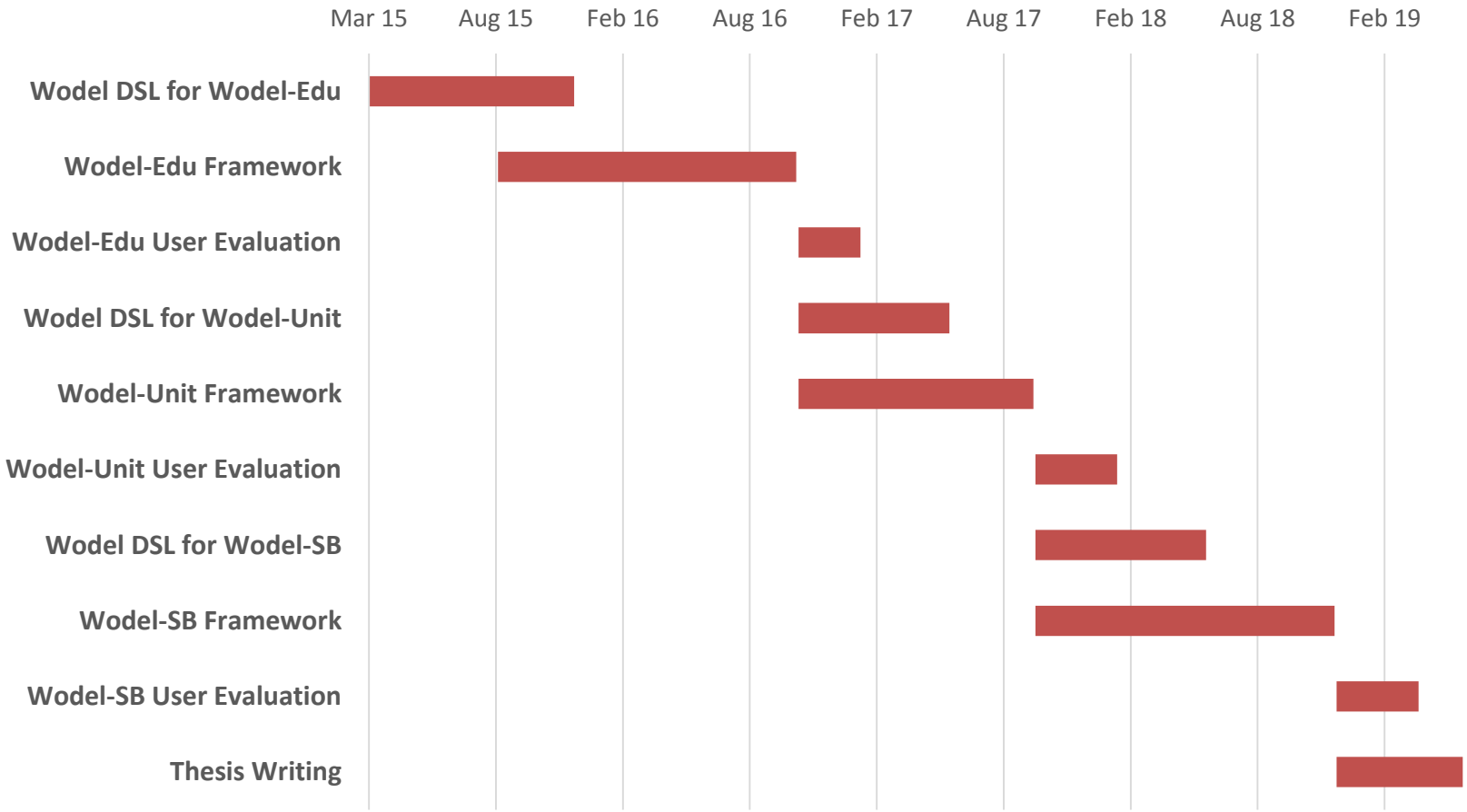
Wodel status

- 7 types of mutation primitives
- 4 selection strategies
- Composite mutations
- Registry extension
- Conditional expressions for specific selection
- Blocks declaration

Wodel-Edu status

- Three kinds of test exercises: alternative response, multiple diagram choice, and multiple emendation choice

PHD. TIMELINE



You are invited to download the source code of this project on GitHub:

http://gomezabajo.github.io/Wodel/

A short video demo of Wodel+Wodel-Edu:

https://youtu.be/T9n3T0jGvzg

Thank you!!

Pablo.GomezA@uam.es