Domain Specific Modeling

Tim Schneider
tim-1.schneider@uni-ulm.de

Institute of Databases and Information Systems, Ulm University

Abstract. *abstract* environment.

1 Introduction

For centuries of people from da Vinci to Einstein have created models to help them better understand patterns and processes in the world around them. Nowadays computers & smartphones make it easier for both novices and domain experts to build and explore their own models and learn new scientific ideas in the process.

- can help experts and novices builing Models for different domains
- enables novices with few or even no programming skills at all to implement their own models (e.g. MindStorm,Starlogo)
- supports domain experts by setting focus on the domain-specific problems (e.g. SimuLink,BPMN)

1.1 Foundation

Model

- What is a model? (Definition [Stachowiak] + some Explanation)
- Abstraction: Remove details which do not serve the purpose
- Homomorphism: Statements on model elements hold for real world entities
- Pragmatics: Model has some purpose
- + example (novice) house Made of LEGOS → Model for a “real” house / building
- + example (expert) blueprints from an architekt → different model, but represents same object in real world
- Abstraction brings representational bias
- → electric cable installations, water pipes cannot be “expressed” in lego but they can with blueprints
- → lego allows 3D Modeling while Blueprints cannot (oonly via 2d projections/ optical illusions)

Domain

- novice and expert have different views on the same thing but use different models / languages
- both models may be used in the context of a domain (Architekture)
- What is a Domain? (Definition + some Explanation)

1.2 Domain Specific Languages

- different approaches : Graphical Modeling Languages, Textual Modeling Languages

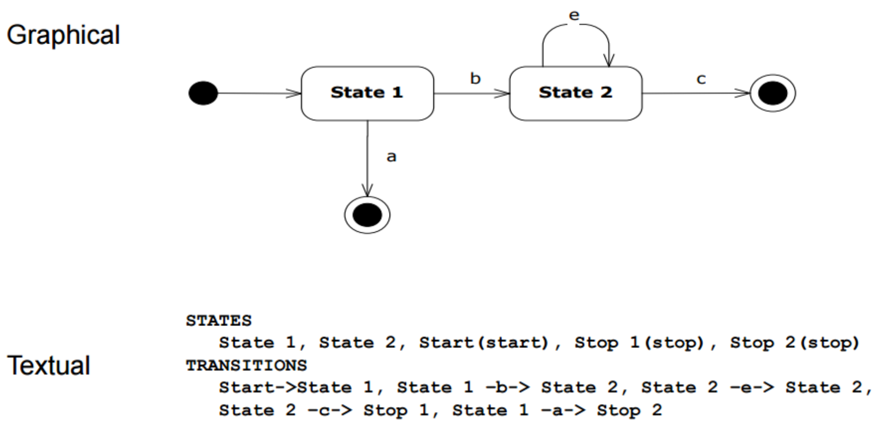

Fig. 1: comparison between textual modeling language and graphcal modeling language for the same model

-
- examples (domain experts): BPMN , SimuLink
- examples (novice): STARLOGO

2 Existing Modeling Languages

2.1 StarLogo TNG

- is a client-based modeling and simulation software
- enables secondary school students and teachers to model decentralized systems through agent-based programming
- facilitates the creation and understanding of simulations of complex systems
- graphical programming blocks instead of text-based computer code

2.2 Snatch

- Graphical Modeling Language (similar to StarLogo)
- developers goal: “make it easy for everyone, of all ages, backgrounds, and interests, to program their own interactive stories, games, animations, and simulations, and share their creations with one another”

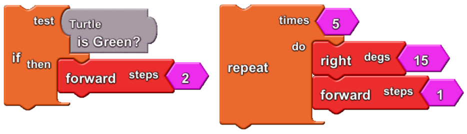

Fig. 2: StarLogo TNGs graphical programming blocks. Example if (left) and repeat (right) blocks are shown. The if block commands a Turtle agent to take two steps forward if it is green; the repeat block commands an agent to repeat five times the sequence of turning right 15 degrees and taking one step forward.

2.3 PhyDSL

- textual modeling for (simple) game dev domain
- based on EMF
- allows codegeneration from the created models
- mobile gameplay definition sections:
  - static actor definition
  - environment and layout definition
  - activities definition
  - scoring rules definition

2.4 Lego Mindstorms

- EV3 Programmer App or Computer Software for programming lego robots in a graphical syntax
- action blocks (Green), flow blocks (Orange), sensor blocks (Yellow), data operation blocks (Red), advanced blocks (Dark blue)
- programms are executed on the EV3 P-brick.

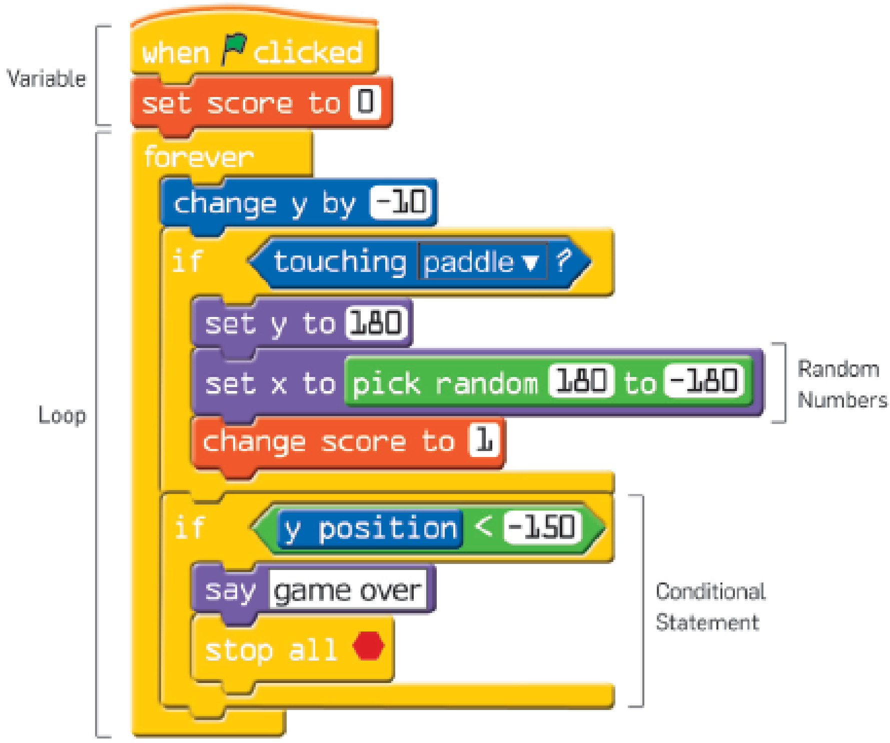

Fig. 3: Sample Scratch script (from Pong-like paddle game) highlighting computational and mathematical concepts

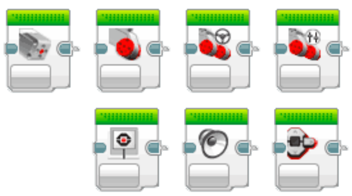

Fig. 7: The action blocks control the actions of the program, e.g. motor rotations, image, sound and the light on the EV3 P-brick.

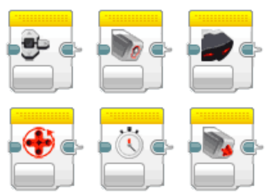

Fig. 8: The sensor blocks allow to read the inputs e.g. from a Color sensor, IR sensor, Touch sensor.

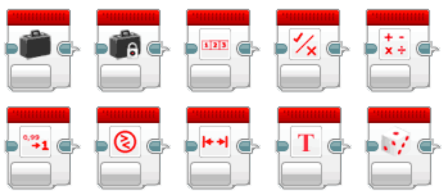

Fig. 9: The data operation blocks let the user write and read variables, compare values for example.

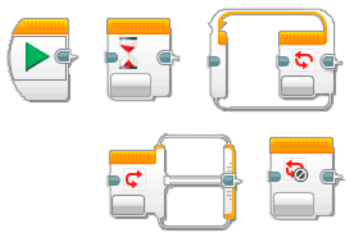

Fig. 10: The Flow blocks control the flow of the program.

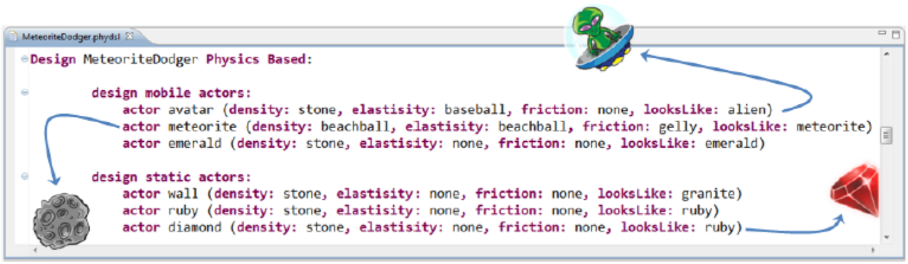

Fig. 4: PhyDSL: static actor definition

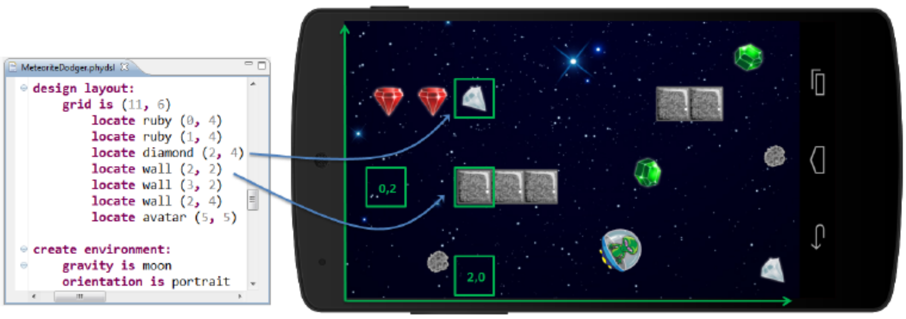

Fig. 5: PhyDSL: environment and layout definition

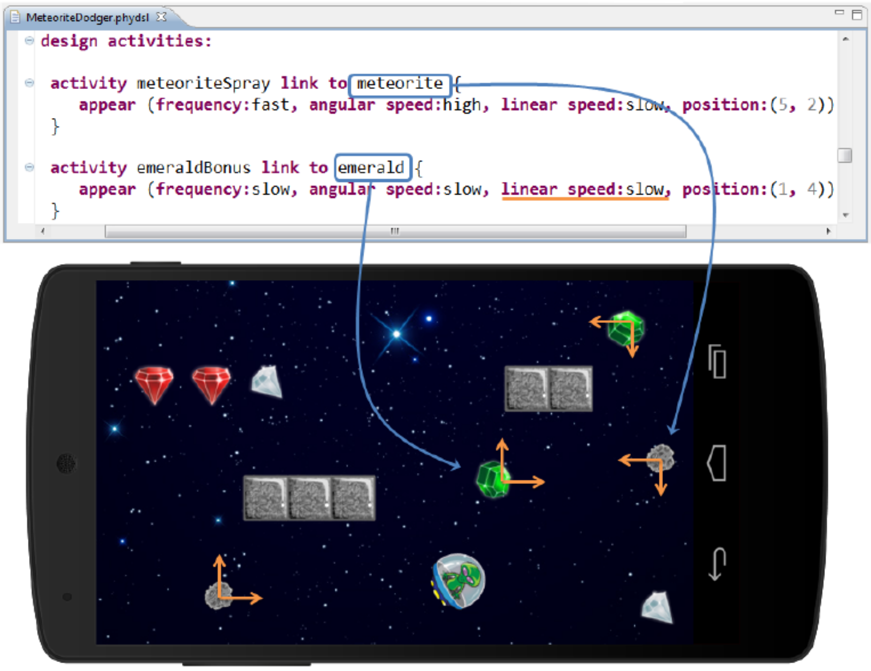

Fig. 6: PhyDSL: activities definition

Fig. 11: EV3 Programmer App used to create programms using a graphical syntax based on blocks

2.5 Sensr

- enables people without programming skills to build mobile data collection and management tools for citizen science

3 Creating Domain Specific Models

3.1 Ecplise EMF

3.2 Ecore

- a meta model for describing models and runtime support for the models .

3.3 Xtext

- used to create textual DSLs for ecore (meta-)models designed in EMF
- syntax similar to EBNF
- one rule for each (meta-)model element

```
FSM: "States:" (states+=State (",")?)*
"Transitions:" (transitions+=Transition (",")?)*;
State: id=ID isStart?="(start)" isStop?="(stop)";
Transition: fromState=[State] "-" (input)? "->"
     toState=[State];
```

Fig. 12: XText Sample Grammar

3.4 GMF

- used to create graphcal DSLs for models described in Ecore
- connects domain model and a graphical model via mapping modell

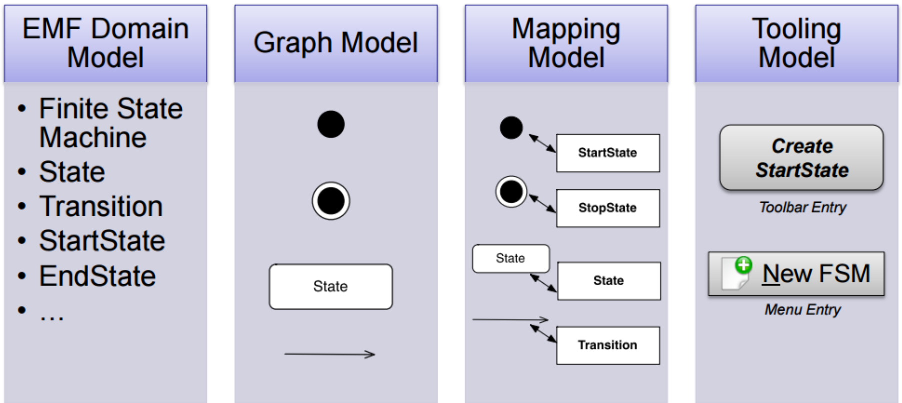

Fig. 13

4 Summary

- DSM allows novices to build and explore their own models and learn new scientific ideas in the process
- domain experts are supported by setting focus on domain specific problems

References

1. OMG: Business Process Management Notation (BPMN) 2.0 (21.05.2010)
2. Smith, V.A., Duncan, I.: Biology students building computer simulations using starlogo tng. Bioscience Education **18**(1) (2011) 1–9
3. Group, L.: Lego mindstorms programming (1999)
4. Amyot, D., Farah, H., Roy, J.F.: Evaluation of development tools for domain-specific modeling languages. In: International Workshop on System Analysis and Modeling, Springer (2006) 183–197
5. Kim, S., Mankoff, J., Paulos, E.: Sensr: evaluating a flexible framework for authoring mobile data-collection tools for citizen science. In: Proceedings of the 2013 conference on Computer supported cooperative work, ACM (2013) 1453–1462
6. Gronback, R.C.: Eclipse modeling project: a domain-specific language (DSL) toolkit. Pearson Education (2009)

7. France, R., Rumpe, B.: Domain specific modeling. Software and Systems Modeling **4**(1) (2005) 1–3
8. Meliones, A., Giannakis, D.: Visual programming of an interactive smart home application using labview. In: 2013 11th IEEE International Conference on Industrial Informatics (INDIN), IEEE (2013) 655–660
9.
10. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., et al.: Scratch: programming for all. Communications of the ACM **52**(11) (2009) 60–67
11. Guana, V., Stroulia, E.: Phydsl: A code-generation environment for 2d physics-based games. In: 2014 IEEE Games, Entertainment, and Media Conference (IEEE GEM). (2014)