Agile Language Engineering

current results and future plans

Tijs van der Storm
storm@cwi.nl / @tvdstorm

CWI
Centrum Wiskunde & Informatica

university of groningen

Inria
INVENTEURS DU MONDE NUMÉRIQUE

DiverSE
Diversity-Centric Software Engineering

CWI SWAT + INRIA DiverSE

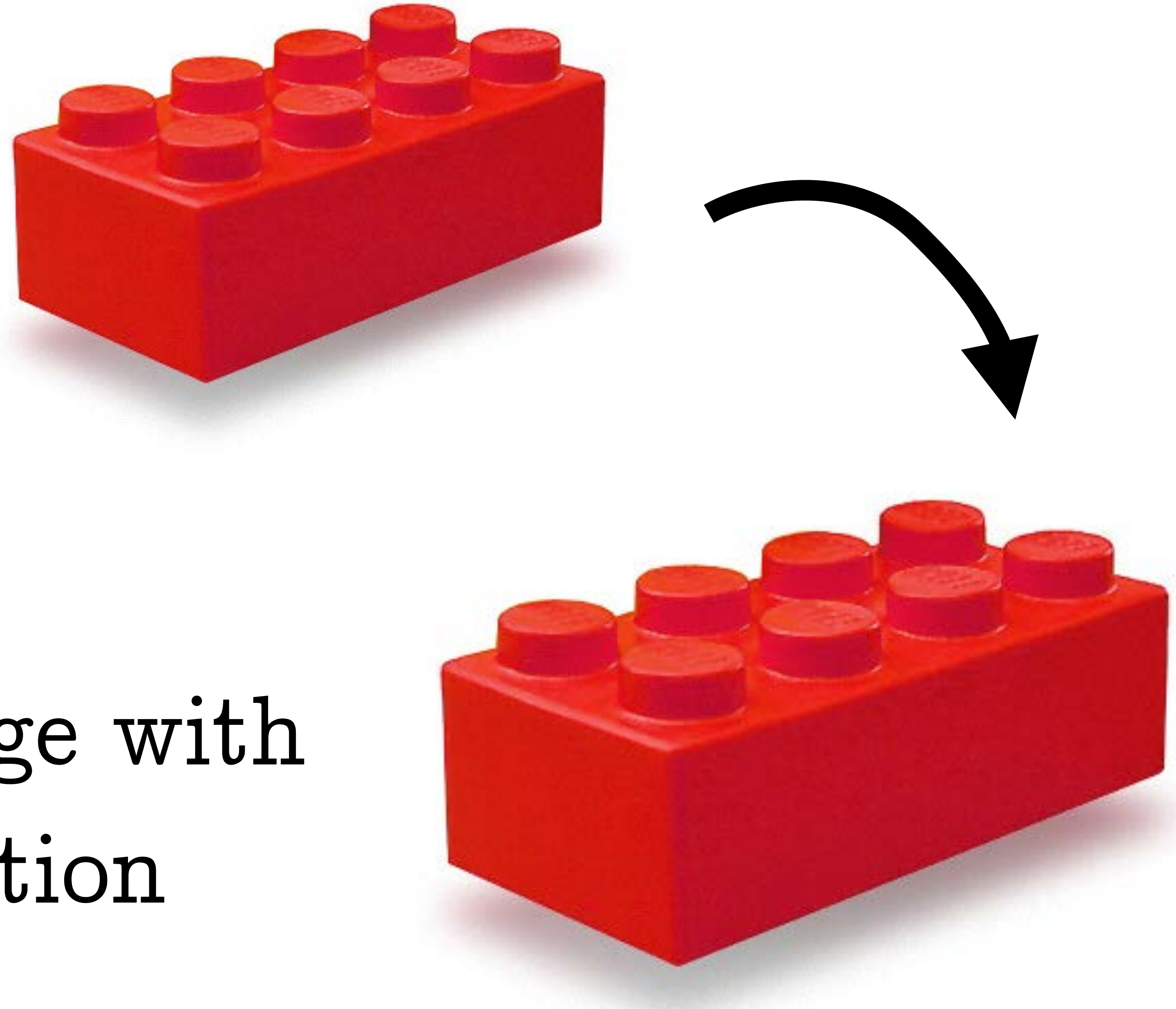
- ALE: Agile Language Engineering
  - increase agility for language **engineers**
  - as well as language **users**
- How:
  - modularity in language engineering
  - live programming for DSLs

Who is/was ALE?

- Benoit Combemale (PI, Inria)
- Olivier Barais (Inria)
- Benoit Baudry (Inria)
- Fabien Coulon (Phd Inria)
- Pierre Jeanjean (Phd Inria)
- Manuel Leduc (Phd Inria)
- Didier Vojtisek (engineer Inria)

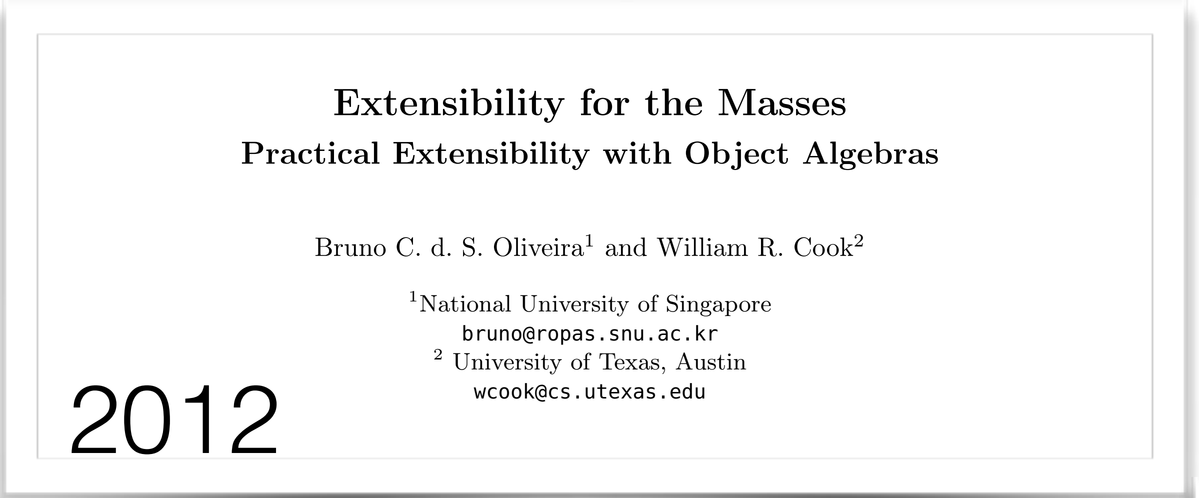
- Tijs van der Storm (co-PI, CWI)
- Thomas Degueule (postdoc CWI, at CNRS per 1-1-‘20)
- Riemer van Rozen (Phd CWI)
- Pablo Inostroza (former Phd CWI, now at SWAT.engineering)
- Ulyana Tikhonova (former post-doc CWI, now in industry)
- Thomas van Binsbergen (Post-doc, CWI)

Modularity



Multiplication feature

Language with addition

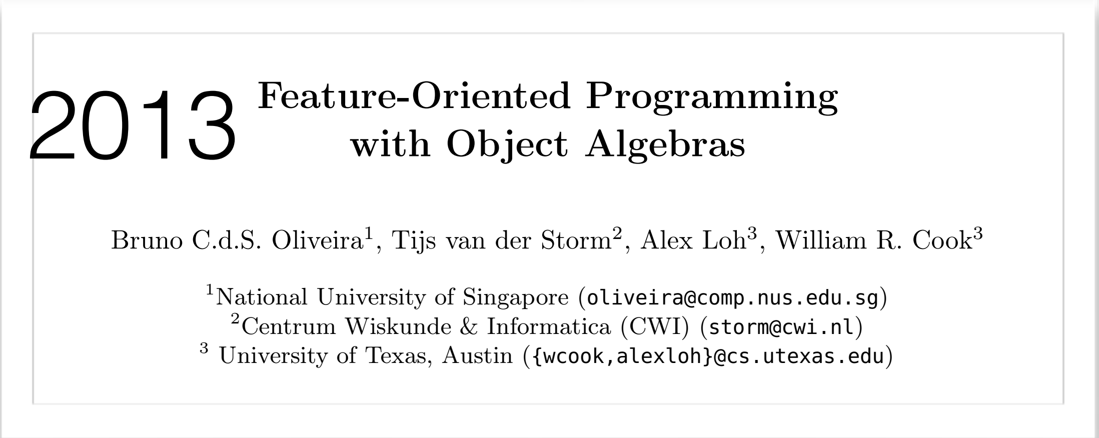


2012

Extensibility for the Masses
Practical Extensibility with Object Algebras

Bruno C. d. S. Oliveira[1] and William R. Cook[2]

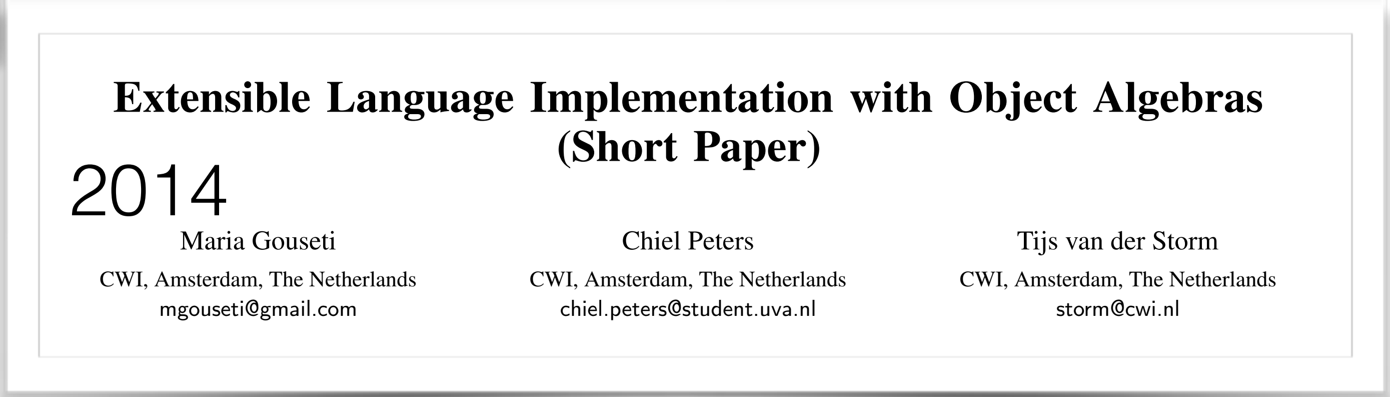
[1]National University of Singapore
bruno@ropas.snu.ac.kr
[2] University of Texas, Austin
wcook@cs.utexas.edu



2013 Feature-Oriented Programming
with Object Algebras

Bruno C.d.S. Oliveira[1], Tijs van der Storm[2], Alex Loh[3], William R. Cook[3]

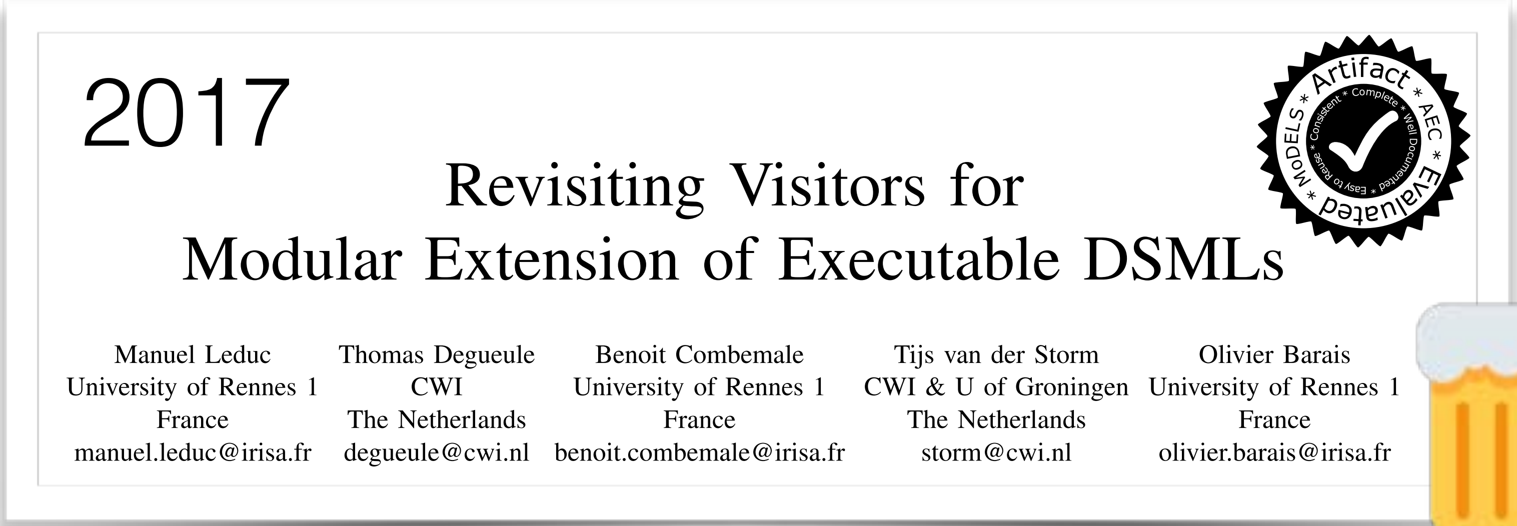
[1]National University of Singapore (oliveira@comp.nus.edu.sg)
[2]Centrum Wiskunde & Informatica (CWI) (storm@cwi.nl)
[3] University of Texas, Austin ({wcook,alexloh}@cs.utexas.edu)



Extensible Language Implementation with Object Algebras
(Short Paper)

2014

Maria Gouseti	Chiel Peters	Tijs van der Storm
CWI, Amsterdam, The Netherlands	CWI, Amsterdam, The Netherlands	CWI, Amsterdam, The Netherlands
mgouseti@gmail.com	chiel.peters@student.uva.nl	storm@cwi.nl

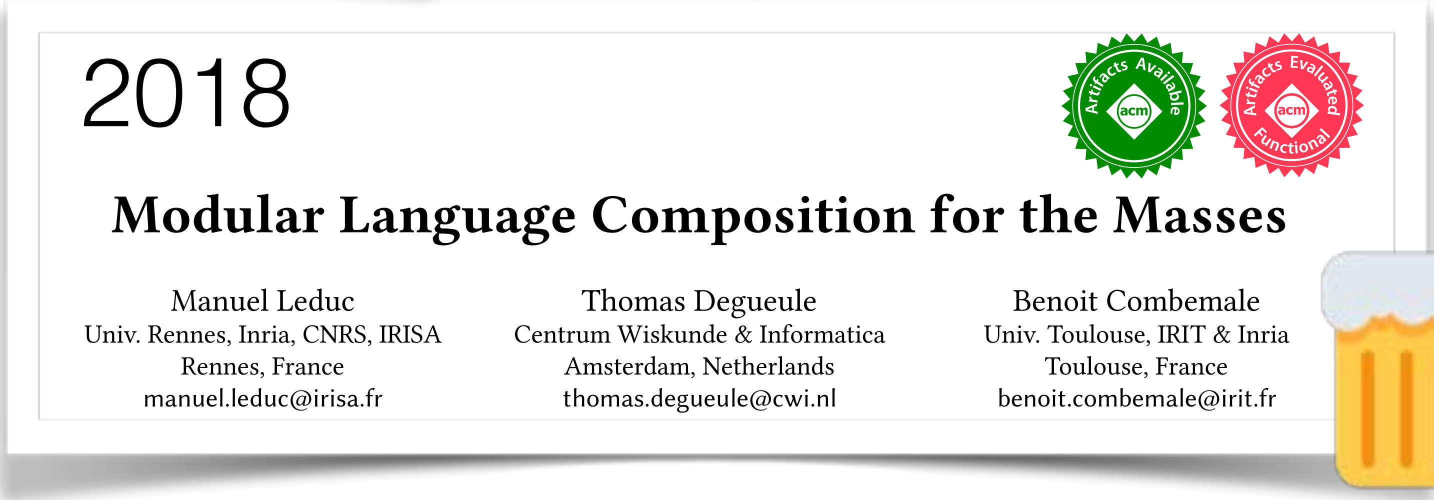


2017

Artifact * AEC * Evaluated * MoDELS *
Consistent * Complete * Well Documented * Easy to Reuse

Revisiting Visitors for
Modular Extension of Executable DSMLs

Manuel Leduc	Thomas Degueule	Benoit Combemale	Tijs van der Storm	Olivier Barais
University of Rennes 1	CWI	University of Rennes 1	CWI & U of Groningen	University of Rennes 1
France	The Netherlands	France	The Netherlands	France
manuel.leduc@irisa.fr	degueule@cwi.nl	benoit.combemale@irisa.fr	storm@cwi.nl	olivier.barais@irisa.fr



2018

Modular Language Composition for the Masses

Manuel Leduc	Thomas Degueule	Benoit Combemale
Univ. Rennes, Inria, CNRS, IRISA	Centrum Wiskunde & Informatica	Univ. Toulouse, IRIT & Inria
Rennes, France	Amsterdam, Netherlands	Toulouse, France
manuel.leduc@irisa.fr	thomas.degueule@cwi.nl	benoit.combemale@irit.fr

SIGPLAN

SPLASH
BOSTON 2018

The prize for the **distinguished artifact** at the
2018 ACM SIGPLAN International Conference on Software Language
Engineering is awarded to

Manuel Leduc, Thomas Degueule and Benoit Combemale

for the artifact associated with the paper

Modular Language Composition for the Masses

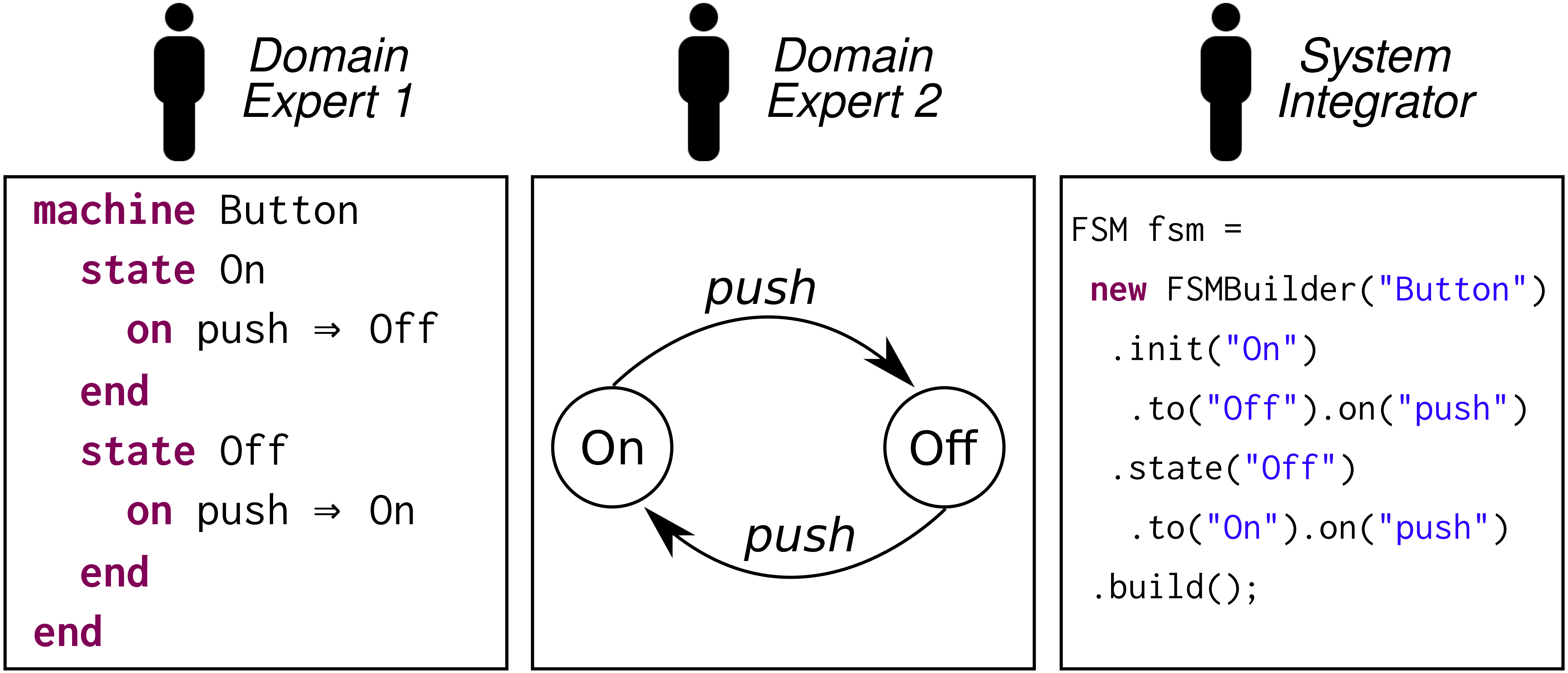
Boston, MA, November 2018

Romina Eramo & Matthew Roberts
(Artifact Evaluation Chairs)

David J. Pearce
(General Chair)

Reuse across technological spaces

- Technological space: modelware, grammarware, ontologyware, …
- All related to language engineering, but different concepts, frameworks, toolkits etc.
- How to make the strengths and tools from different spaces available to each other?



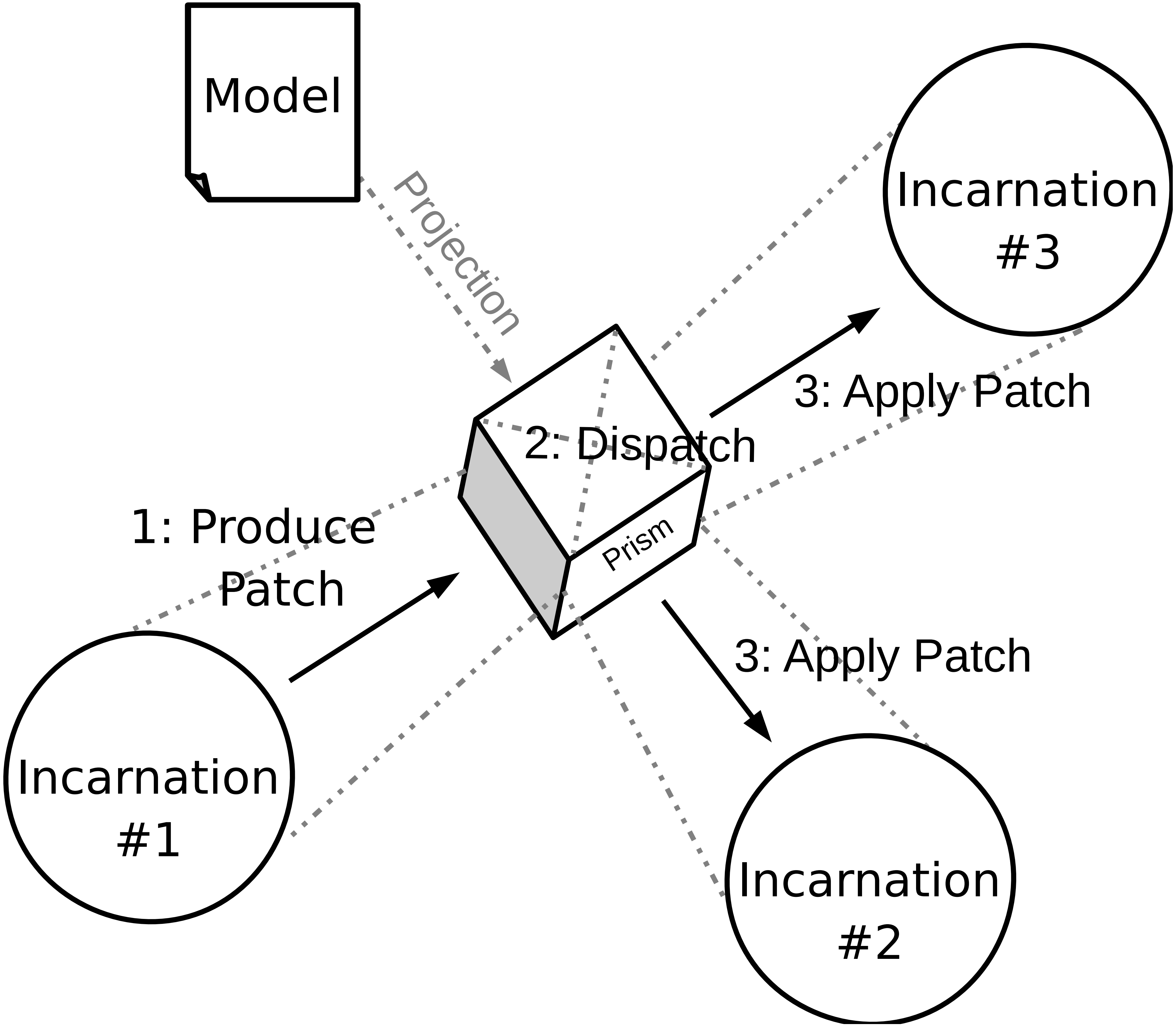
Domain Expert 1

```
machine Button
  state On
    on push ⇒ Off
  end
  state Off
    on push ⇒ On
  end
end
```

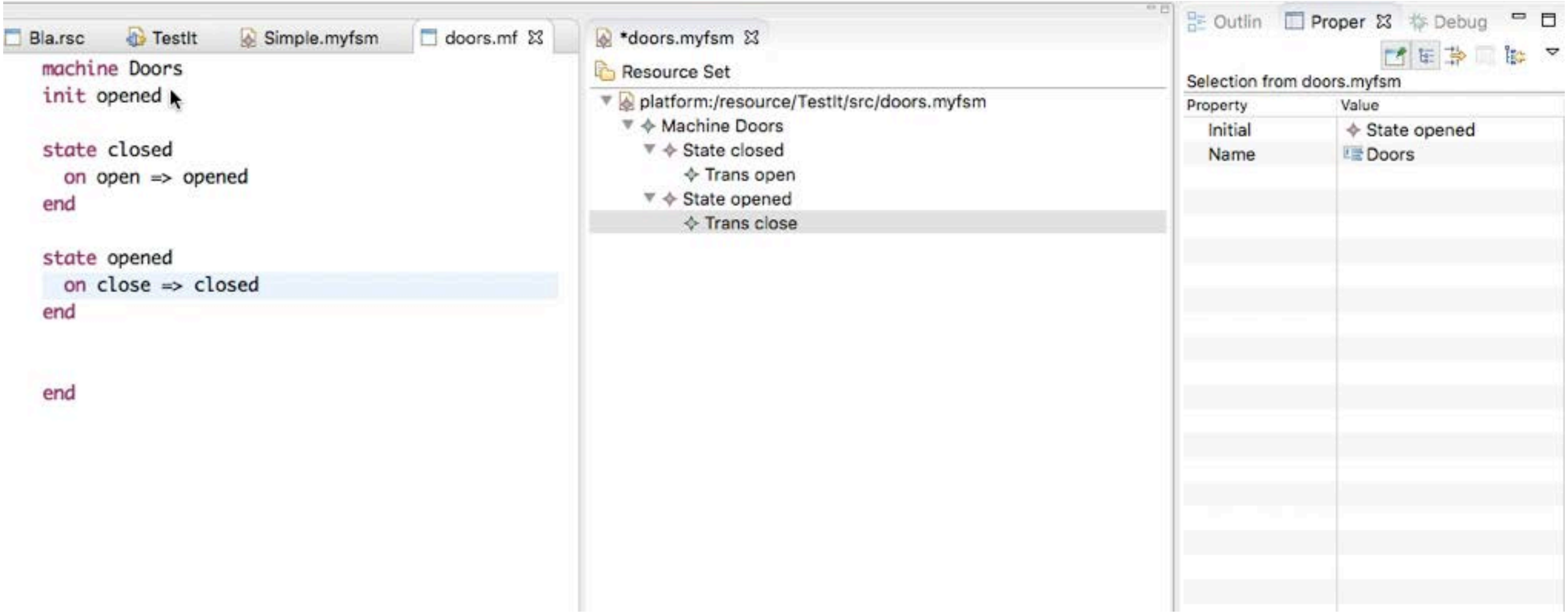
Domain Expert 2

System Integrator

```
FSM fsm =
 new FSMBuilder("Button")
  .init("On")
   .to("Off").on("push")
  .state("Off")
   .to("On").on("push")
 .build();
```


Model
Projection
Incarnation #3
3: Apply Patch
2: Dispatch
Prism
1: Produce Patch
3: Apply Patch
Incarnation #1
Incarnation #2

Synchronizing shapes



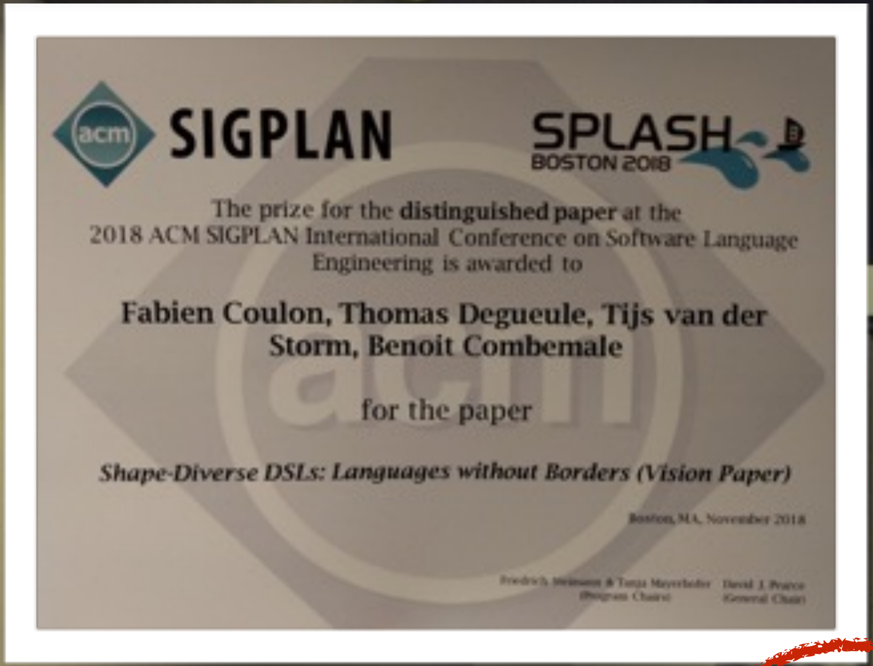
```
machine Doors
init opened

state closed
  on open => opened
end

state opened
  on close => closed
end


end
```

Resource Set

```
▼ platform:/resource/TestIt/src/doors.myfsm
  ▼ Machine Doors
    ▼ State closed
        Trans open
    ▼ State opened
        Trans close
```

Selection from doors.myfsm

Property	Value
Initial	State opened
Name	Doors



SIGPLAN

SPLASH
BOSTON 2018

The prize for the **distinguished paper** at the
2018 ACM SIGPLAN International Conference on Software Language
Engineering is awarded to

**Fabien Coulon, Thomas Degueule, Tijs van der
Storm, Benoit Combemale**

for the paper

Shape-Diverse DSLs: Languages without Borders (Vision Paper)

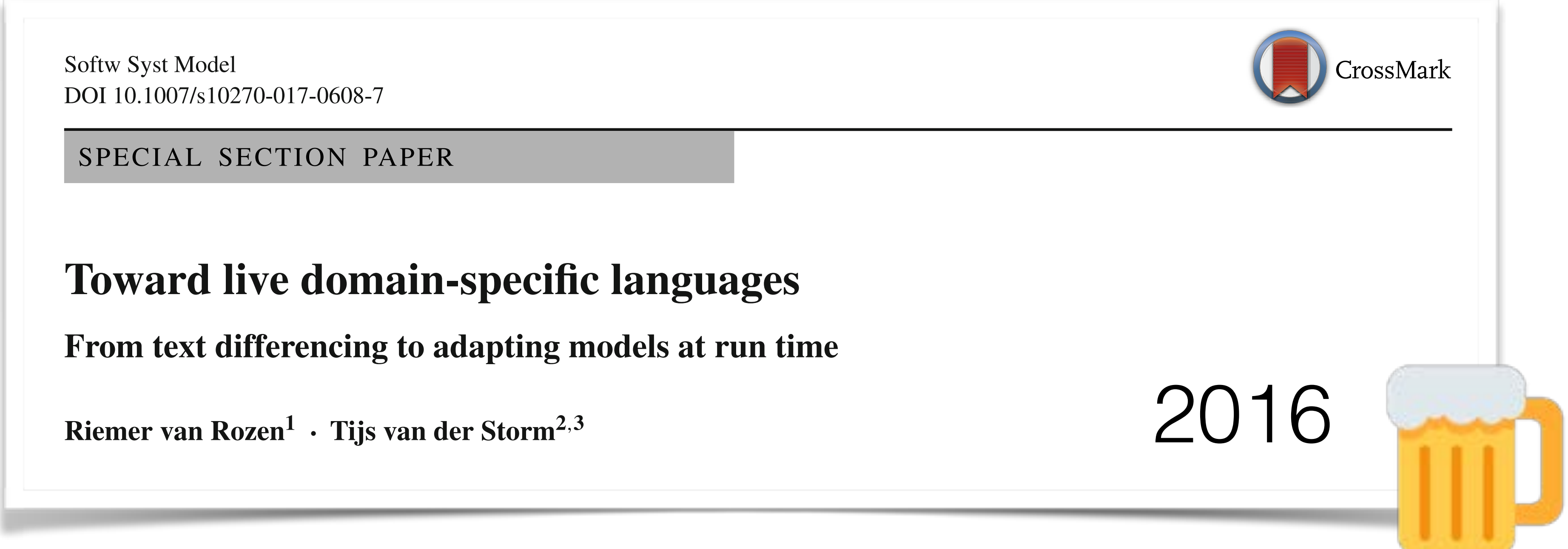
Boston, MA, November 2018

Friedrich Steimann & Tanja Mayerhofer
(Program Chairs)

David J. Pearce
(General Chair)

Live modeling

- Live programming: make programming experience more fluid by getting rid of slow edit-compile-run cycle
- Live modeling: do the same for domain-specific modeling languages
- Question: when the language user changes the program, how should we migrate the run-time state?



Softw Syst Model
DOI 10.1007/s10270-017-0608-7

CrossMark

SPECIAL SECTION PAPER

Toward live domain-specific languages

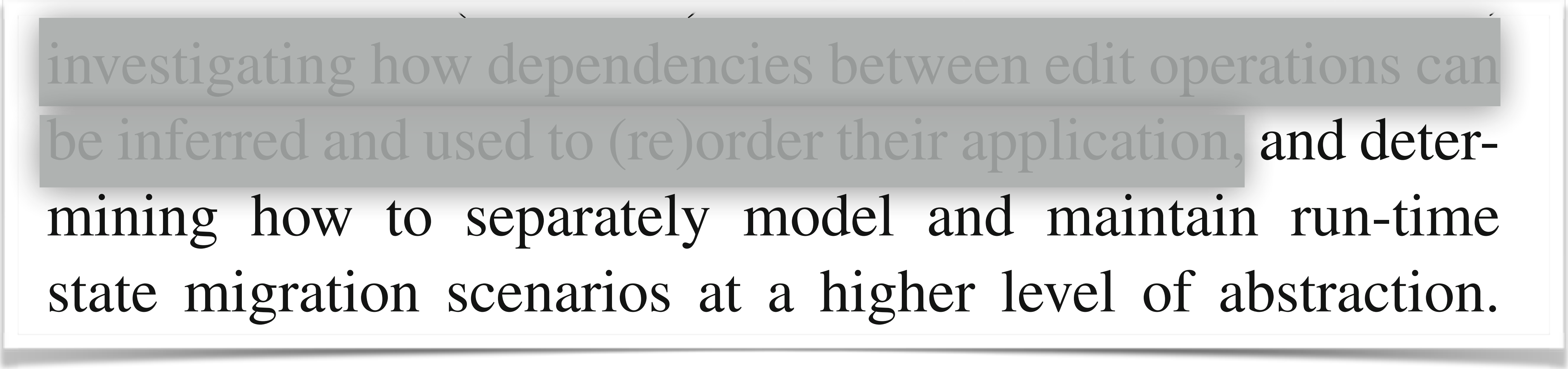
From text differencing to adapting models at run time

Riemer van Rozen[1] · Tijs van der Storm[2,3]

2016

Implementing run-time state migration manually, is language-specific, tedious and error-prone

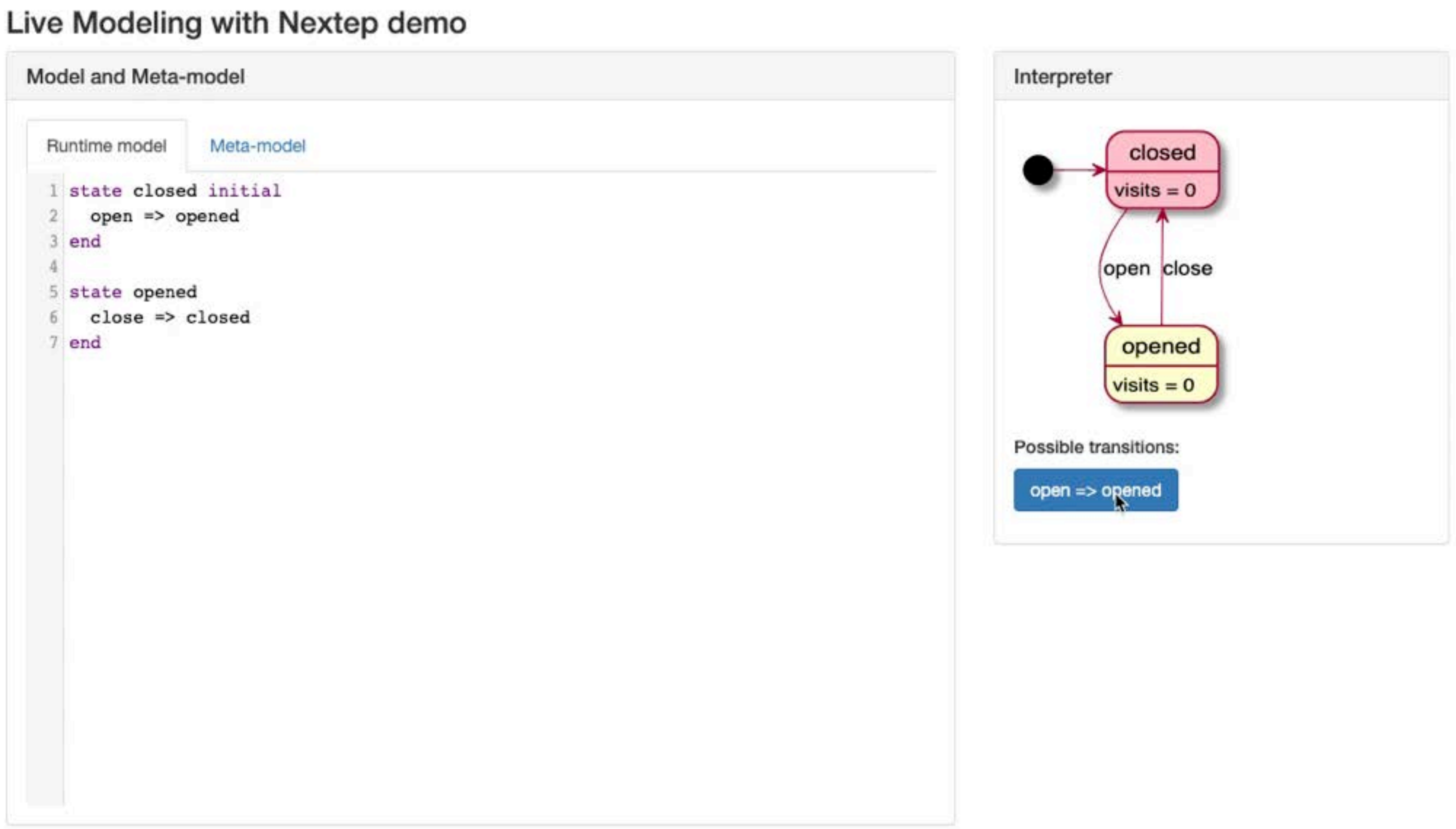
Future work



investigating how dependencies between edit operations can be inferred and used to (re)order their application, and determining how to separately model and maintain run-time state migration scenarios at a higher level of abstraction.

Nextep: constraint-based run-time state migration

- Describe the relation between the abstract syntax of the language and its run-time state structure using **constraints**
- When a program is edited, use **constraint solving** to find a new run-time state compatible with the new version



Live Modeling with Nextep demo

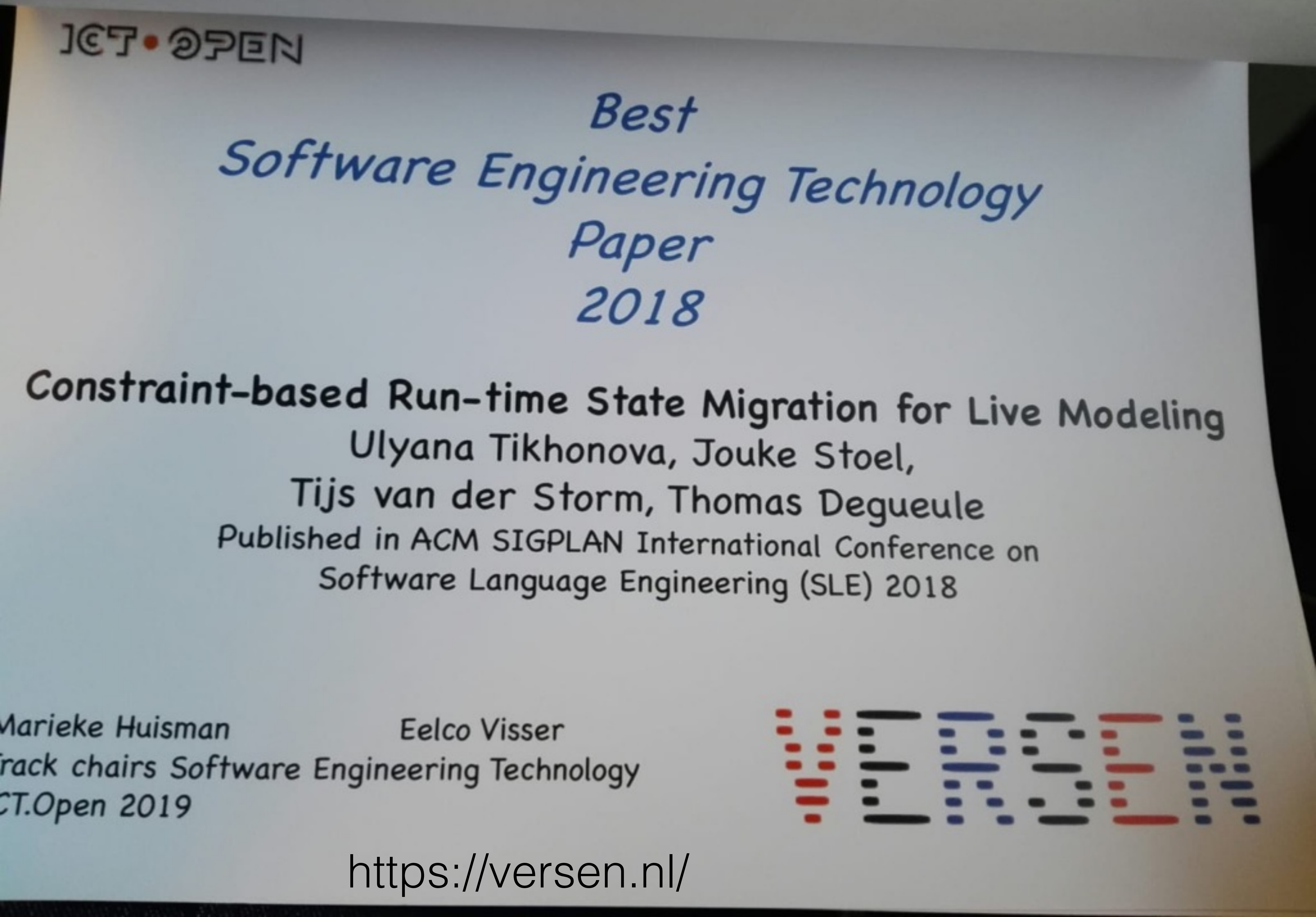
Model and Meta-model

Runtime model | Meta-model

```
state closed initial
  open => opened
end

state opened
  close => closed
end
```

Interpreter

Possible transitions:



ICT•OPEN

Best
Software Engineering Technology
Paper
2018

Constraint-based Run-time State Migration for Live Modeling

Ulyana Tikhonova, Jouke Stoel,
Tijs van der Storm, Thomas Degueule

Published in ACM SIGPLAN International Conference on
Software Language Engineering (SLE) 2018

Marieke Huisman Eelco Visser
rack chairs Software Engineering Technology
CT.Open 2019

VERSEN

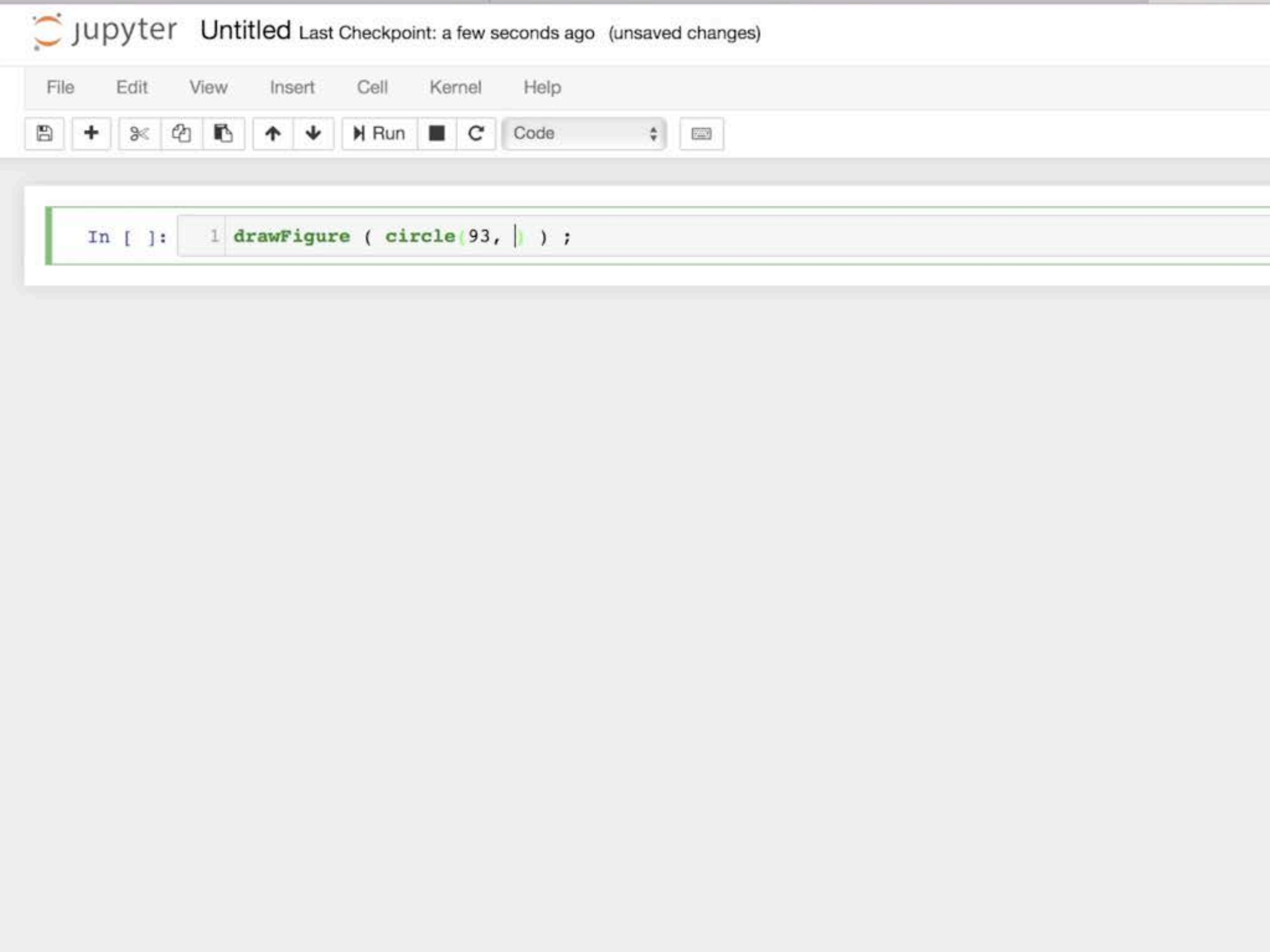
https://versen.nl/

Future of ALE

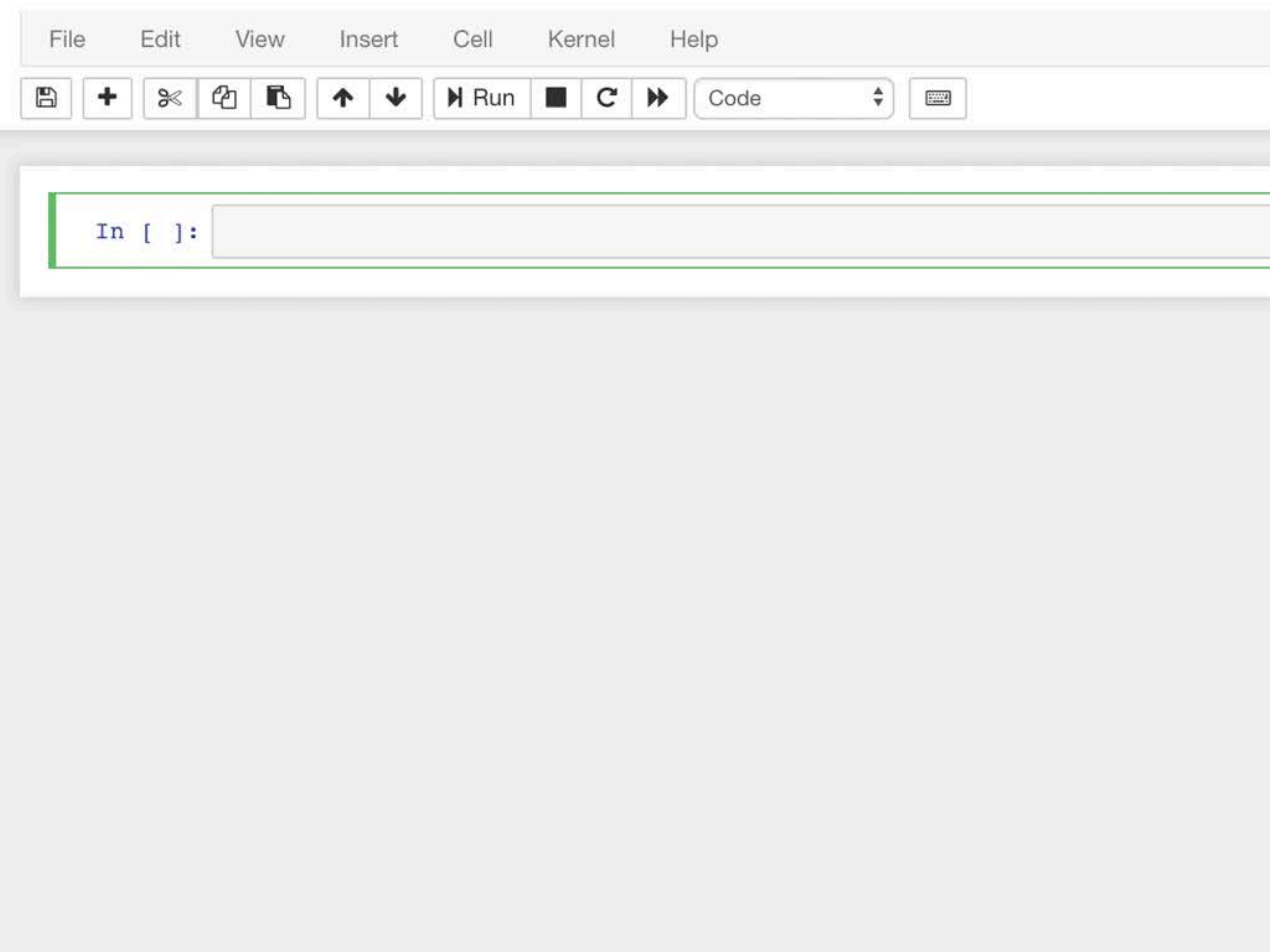
- Applying for new term 2020-2023
- Continue in the same research line
- New topics to explore:
  - Live computational notebooks
  - Collaborative live modeling

Towards Live Notebooks

- Notebooks: popular style of literate programming
  - data science, scientific programming
  - think: Mathematica
- Generic techniques for constructing notebooks for arbitrary DSLs
- Making notebooks “live”


jupyter
Untitled Last Checkpoint: a few seconds ago (unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Help
Run
Code
In [ ]:

```
drawFigure ( circle(93, ) ) ;
```


File
Edit
View
Insert
Cell
Kernel
Help
Run
Code
In [ ]:

Agile Language Engineering

- Productive collaboration between CWI SWAT and INRIA DiverSE
- Improving language engineering and language use in the context of domain-specific languages
- 5 publications, numerous workshops, 3 awards.
- On to version 2.0!

http://gemoc.org/ale/