DSL Composition for Model-Based Test Generation

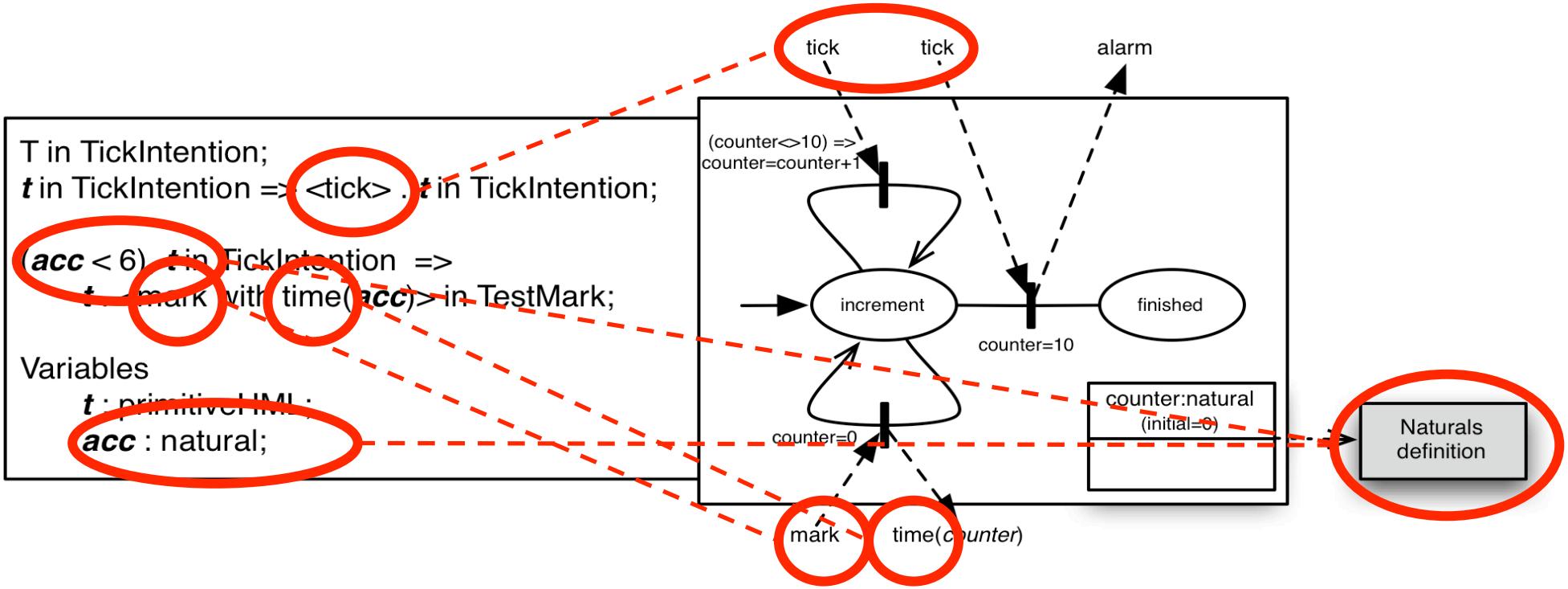
(or Adding Testability to a DSL by using DSL Composition)

Bruno Barroca, Vasco Amaral and Luís Pedro

Levi Lúcio, Didier Buchs

UNIVERSITÉ DE GENÈVE

Merge of SATEL and HALL



What is SATEL?

- Template language for test generation

- Depends on the
  - Syntax of the target DSL: input and output types of the target DSL and data type signatures
  - Semantics of the target DSL: step and type semantics of the target DSL

Semantics of the Merged Language

HALL model $\models$ *Instantiated Test Intention ?*

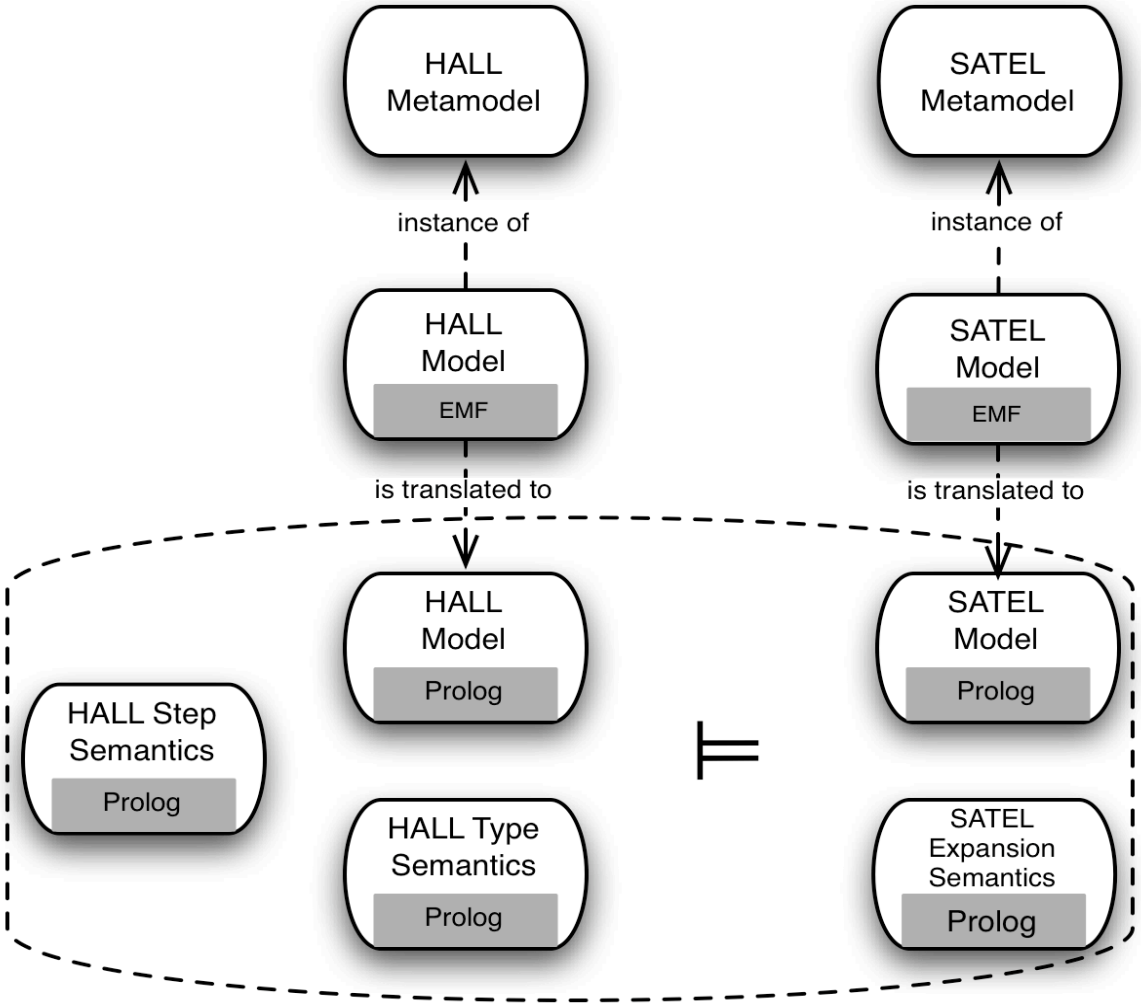
<tick> <mark with time(1)>, true
<tick> <tick> <mark with time(2)>, true
…

Valid tests for *TestMark*

<tick> <mark with time(2)>, false
<tick> <tick> <mark with time(1)>, false
…

Invalid tests for *TestMark*

Starting point: merge of SATEL and HALL in Prolog



Motivation

We **merged**, using Prolog, a test specification language and a DSL to add testability to that DSL.

Can we formalize that merge in terms of **language composition**?

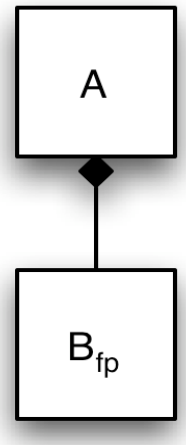
Can we use the formalization to add testability to any DSL?

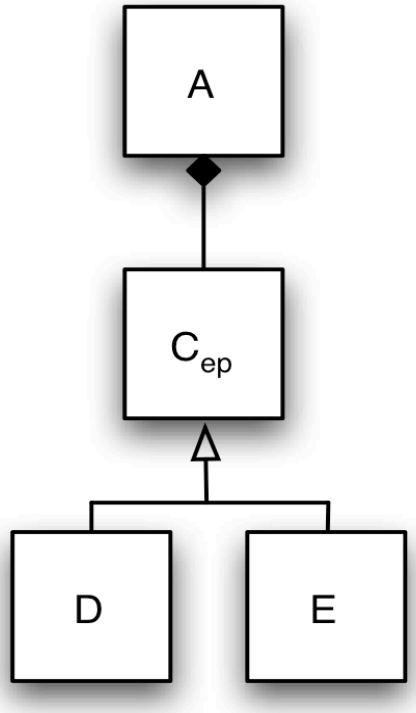
Road Map

- Starting point: merge of SATEL and HALL in Prolog
- **CoPsy: a DSL composition framework**
- SATEL + HALL composition formalization
- Extrapolation of the approach
- Conclusions and future work

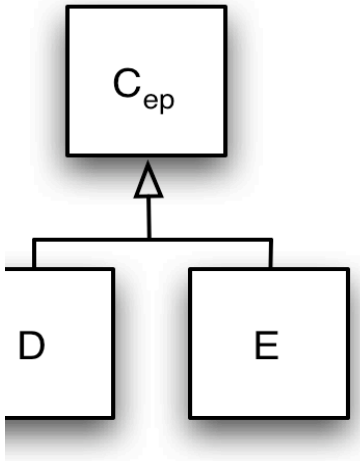
CoPsy: Syntactic Composition

parametric metamodel

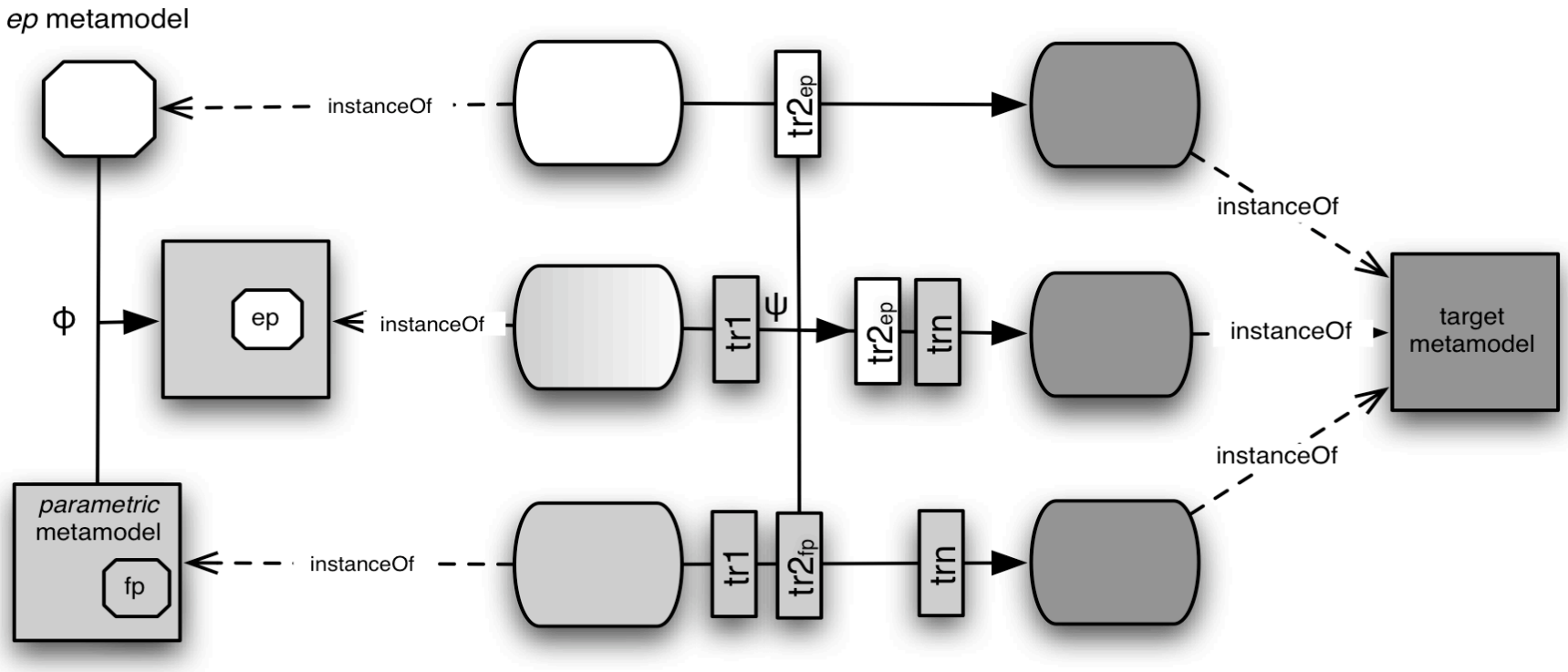




effective parameter metamodel



CoPsy: Semantic Composition

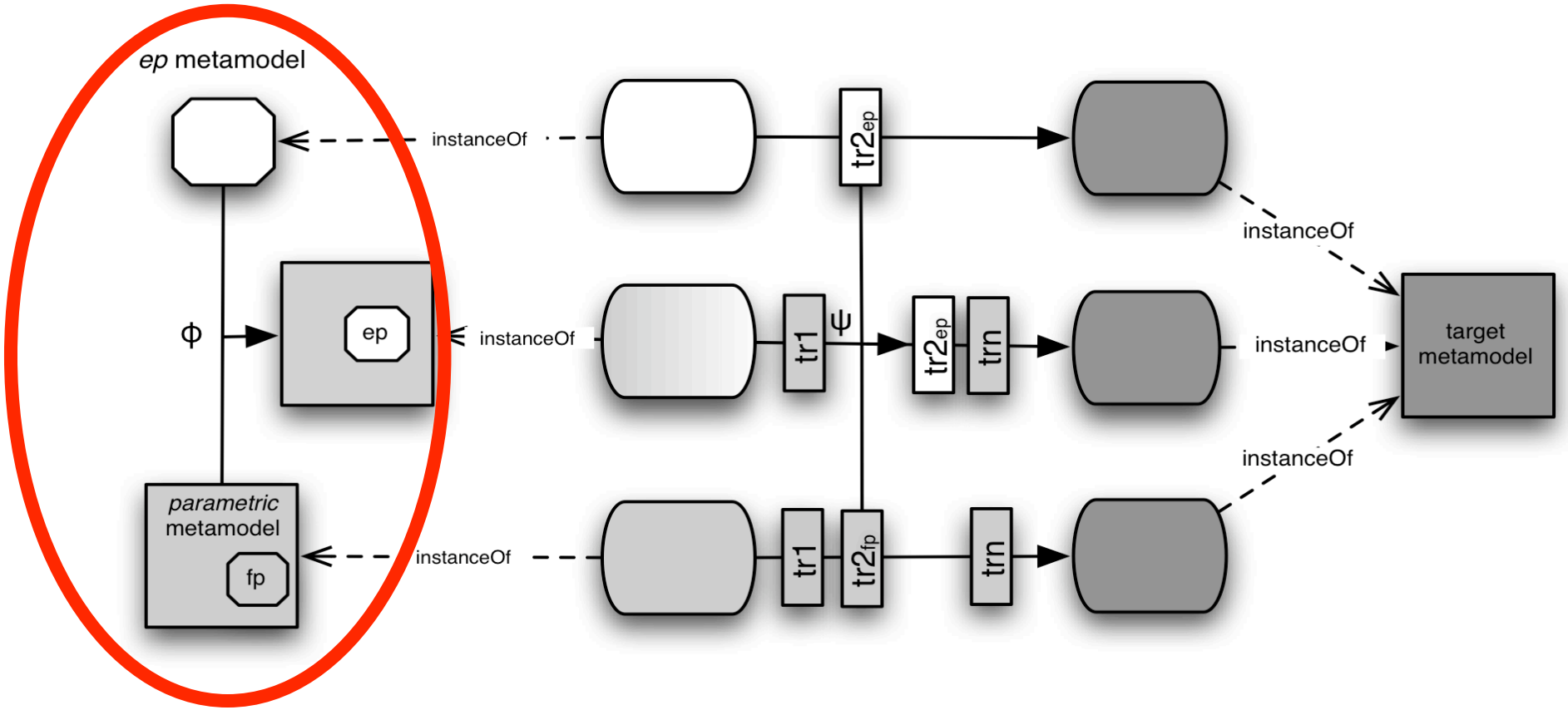


$\Psi=\{(tr2_{fp},tr2_{ep})\}$

Road Map

- Starting point: merge of SATEL and HALL in Prolog
- CoPsy: a DSL composition framework
- **SATEL + HALL composition formalization**
- Extrapolation of the approach
- Conclusions and future work

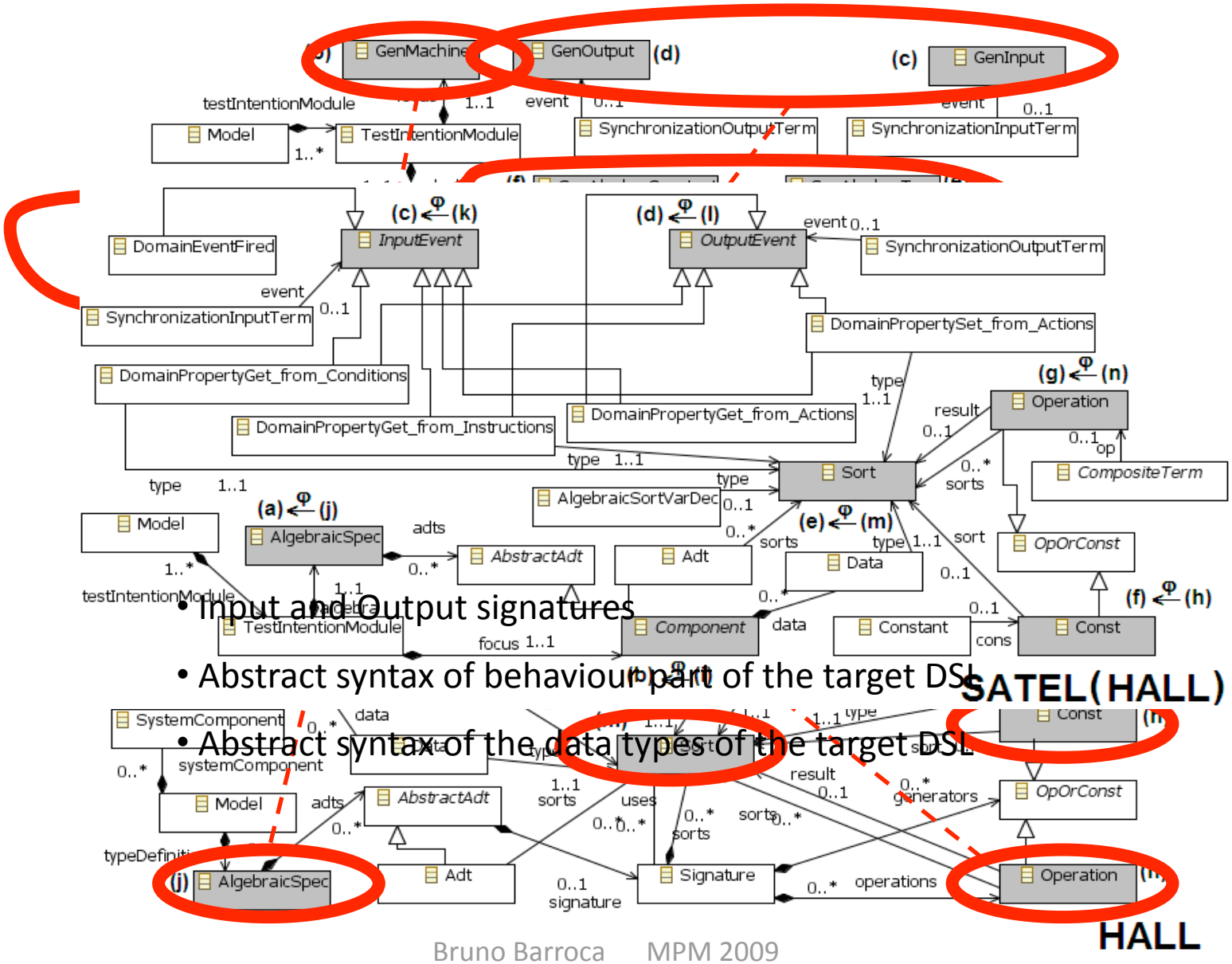
Composition Map



Syntactic Composition

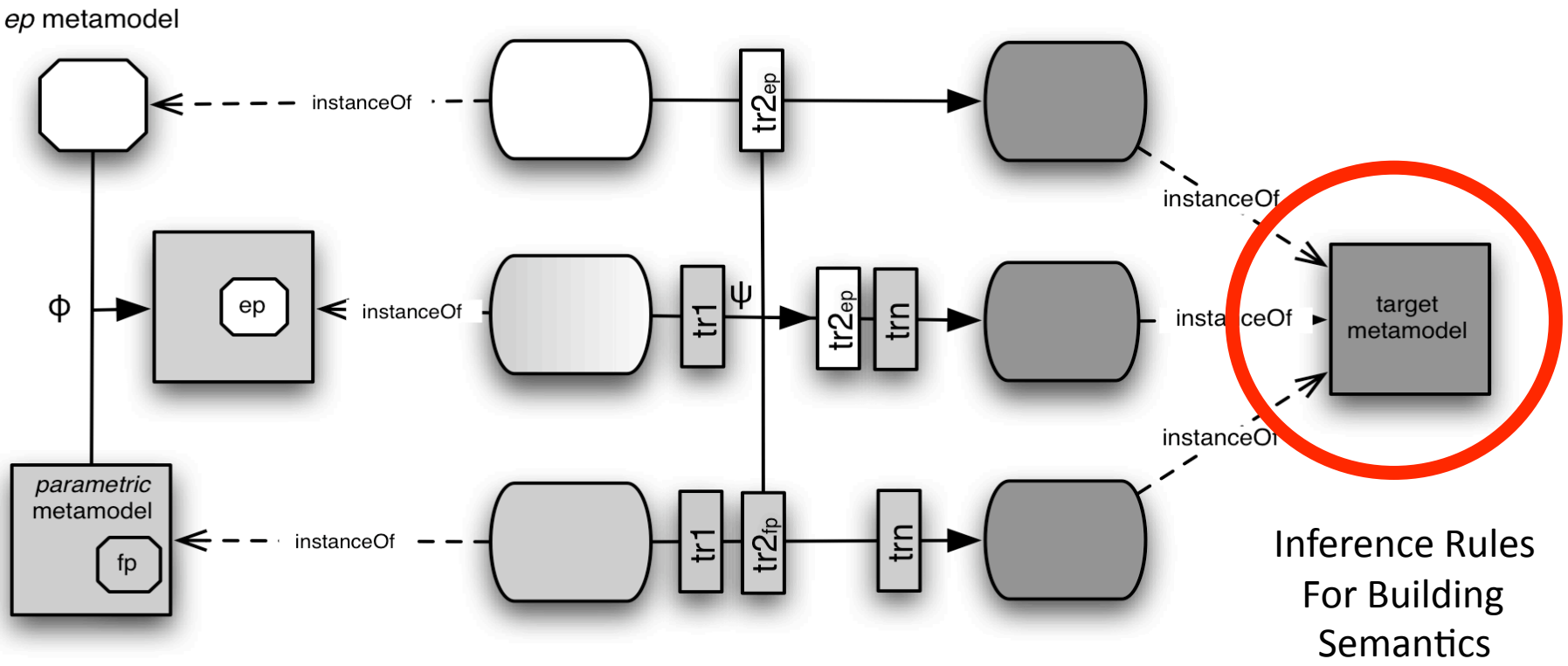
Syntactic Composition

Φ= {(a,j),
(b,i),
(c,k),
(d,l),
(e,m),
(g,n),
(f,g)}

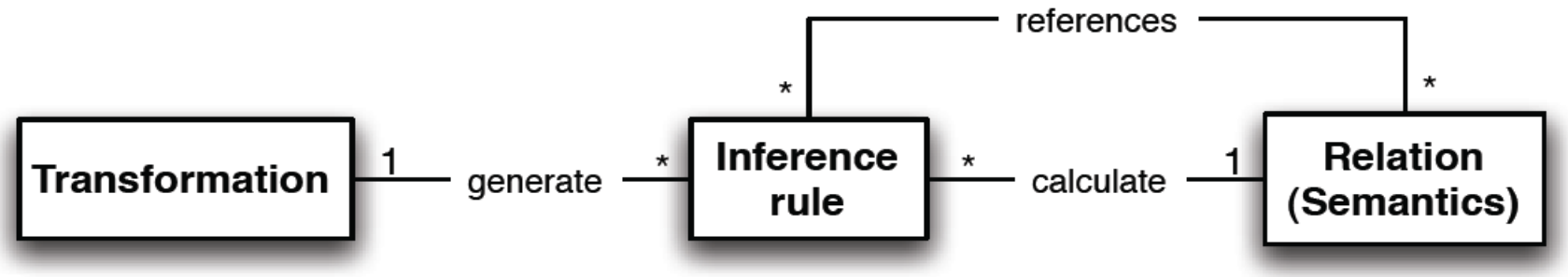


- Input and Output signatures
- Abstract syntax of behaviour part of the target DSL
- Abstract syntax of the data types of the target DSL

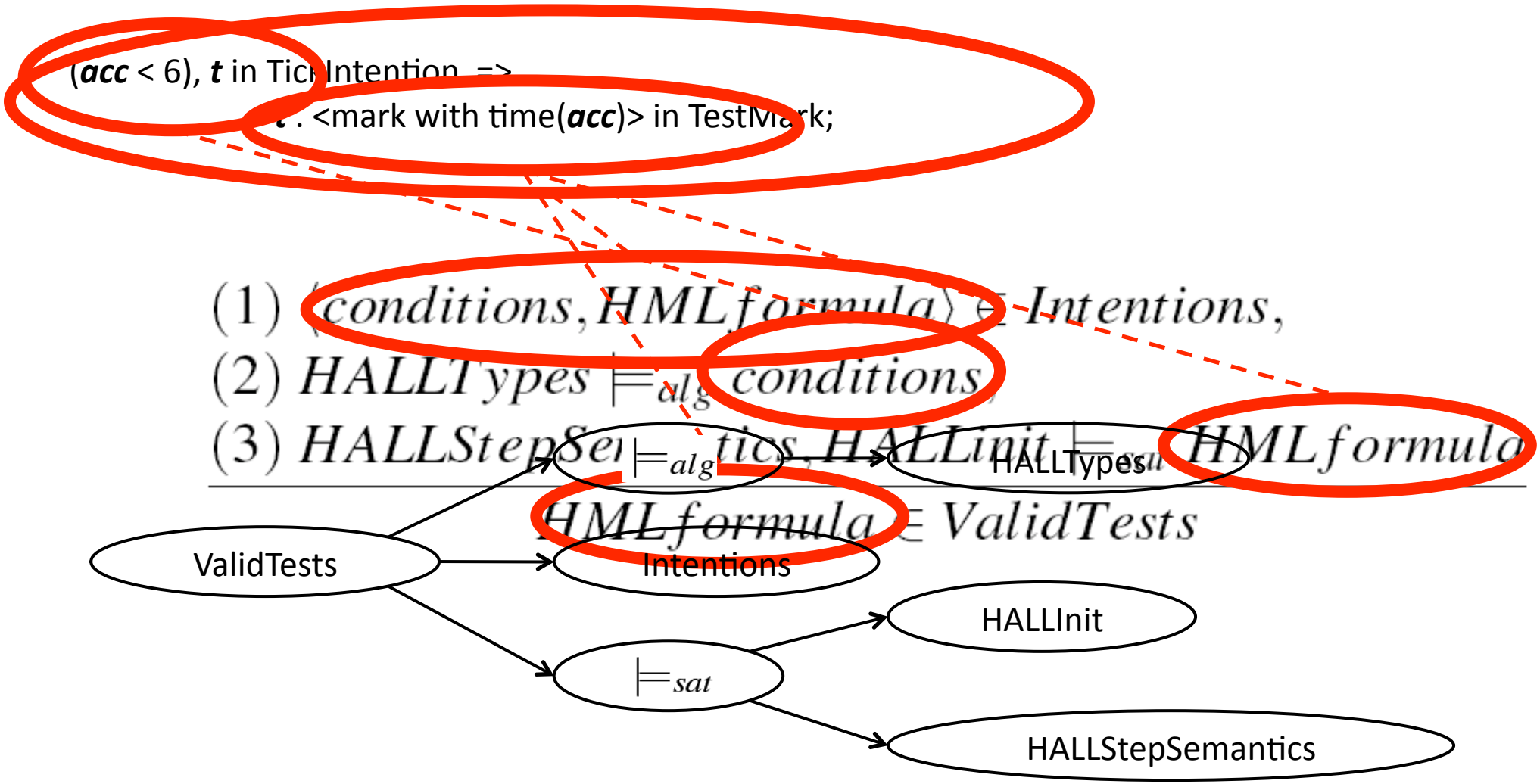
Composition Map



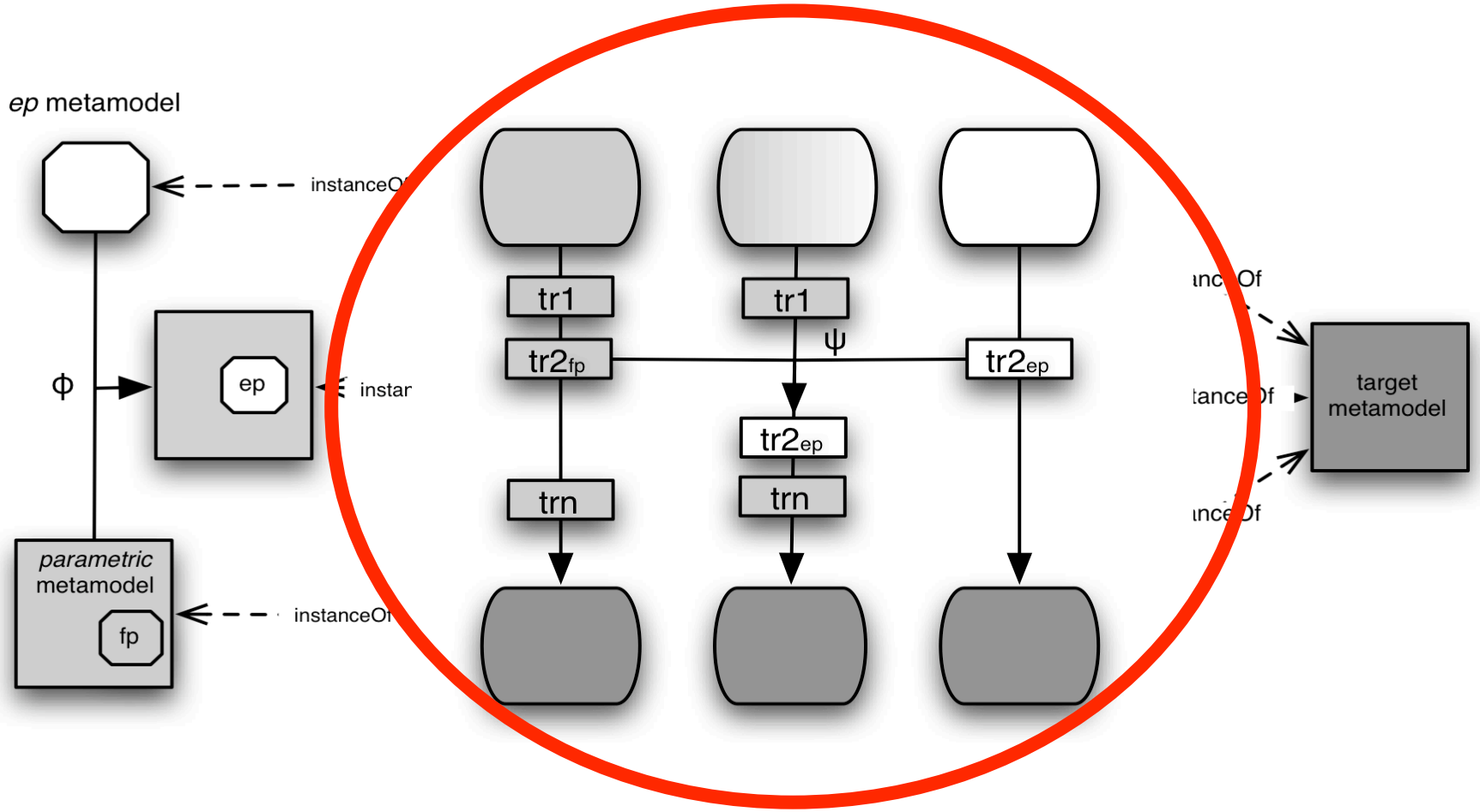
Inference Rules for Building Semantics



Inference Rule Example



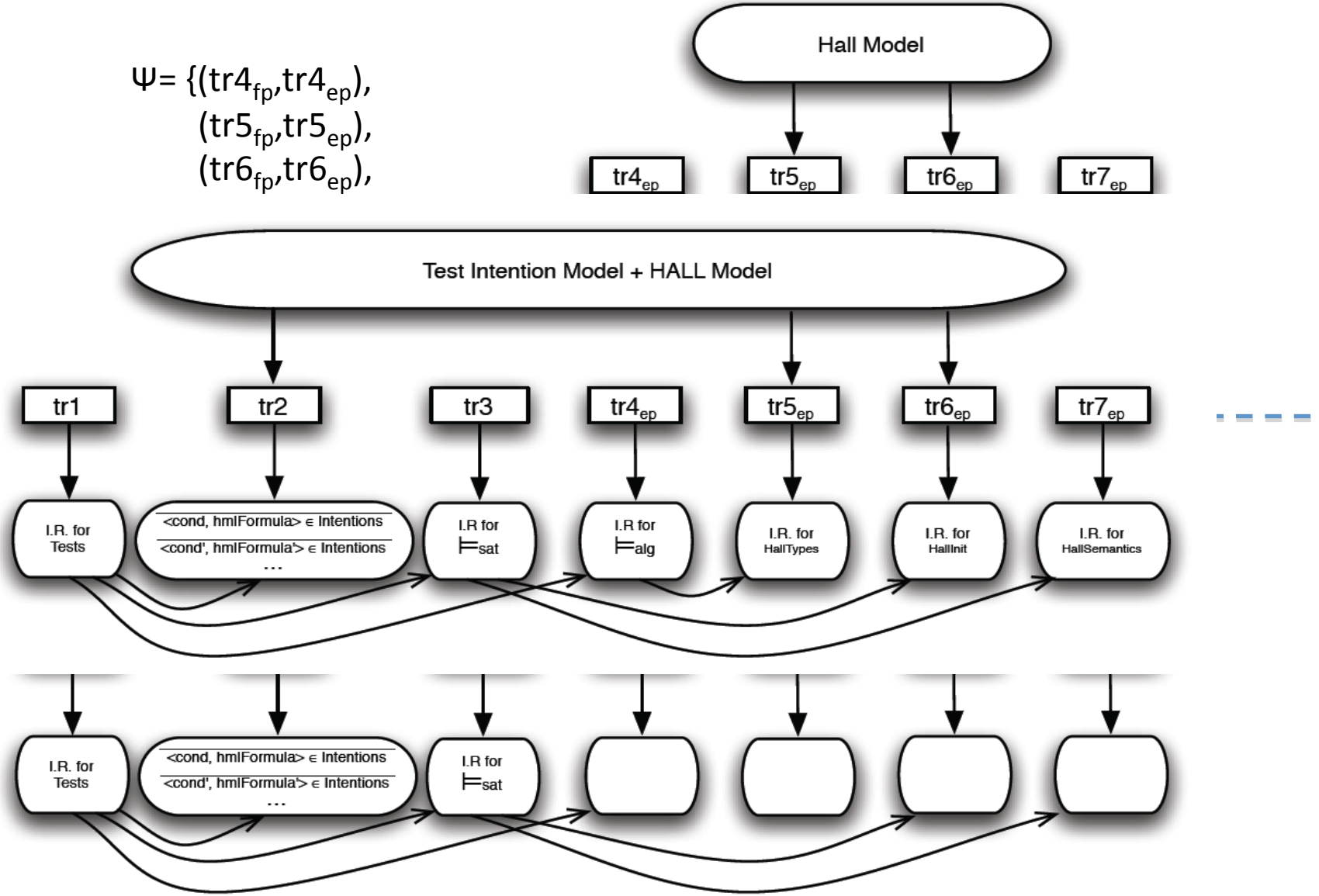
Composition Map



Transformation Composition

Transformation Composition (partial)

$\Psi = \{(tr4_{fp}, tr4_{ep}),$
$(tr5_{fp}, tr5_{ep}),$
$(tr6_{fp}, tr6_{ep}),$



Road Map

- Starting point: merge of SATEL and HALL in Prolog
- CoPsy: a DSL composition framework
- SATEL + HALL composition formalization
- **Extrapolation of the approach**
- Conclusions and future work

Extrapolation of the approach

- Using *CoPsy*, we need to identify, in the target DSL, the abstract syntax for:
  - Inputs/outputs
  - Operational behaviour specification
  - Data types specification
- And the transformation rules to generate the inference rules to compute for the target DSL:
  - The step semantics
  - The semantics of types

Conclusions

- We merged a test specification language (SATEL) with component specification language (HALL)
- We formalized a SATEL + HALL merge in terms of a **language composition**.

Future Work

- Can we use that formalization to add testability to any DSL?
  - We are now implementing the composition approach;
  - work in the systematization of the approach in a meta-modelling workbench.
  - In order to validate the extrapolation we will apply it to another DSL (APN).

Thank you!

Questions?