Towards Self-Adaptable Languages

Gwendal Jouneaux [1] Olivier Barais [1]
Benoit Combemale [1] Gunter Mussbacher [2]

[1]Univ. Rennes, Inria, IRISA – Rennes, France

[2]McGill University – Montreal, Canada

CCI Team talk — October 21, 2022

Context

Software ...

Context

Software ...

- Evolve in complex/changing environment (e.g, Cloud, embedded systems)

Context

Software ...

- Evolve in complex/changing environment (e.g, Cloud, embedded systems)
- Need dynamic adaptation to best deliver the service (e.g., Waymo[1], Netflix[1])

[1] Cf. `https://waymo.com`, `https://www.netflix.com`

Context

Software ...

- Evolve in complex/changing environment (e.g, Cloud, embedded systems)
- Need dynamic adaptation to best deliver the service (e.g., Waymo[1], Netflix[1])

Software languages ...

[1] Cf. `https://waymo.com`, `https://www.netflix.com`

Context

Software ...

- Evolve in complex/changing environment (e.g, Cloud, embedded systems)
- Need dynamic adaptation to best deliver the service (e.g., Waymo[1], Netflix[1])

Software languages ...

- Can abstract concerns into high level constructs (e.g., memory management)

[1] Cf. `https://waymo.com`, `https://www.netflix.com`

Context

Software ...

- Evolve in complex/changing environment (e.g, Cloud, embedded systems)
- Need dynamic adaptation to best deliver the service (e.g., Waymo[1], Netflix[1])

Software languages ...

- Can abstract concerns into high level constructs (e.g., memory management)

Vision : abstract self-adaption into high level language constructs

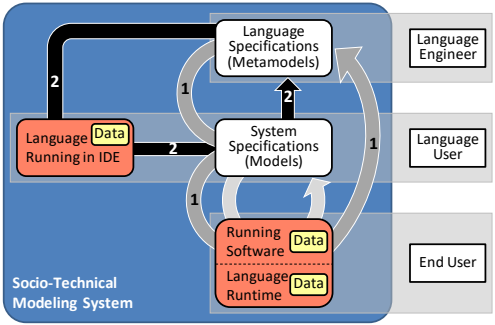
[1] Cf. `https://waymo.com`, `https://www.netflix.com`

What is a Self-Adaptable Language ?

" *A software language that abstracts the design and execution of feedback loops in the design-time environment and the run-time environment* "

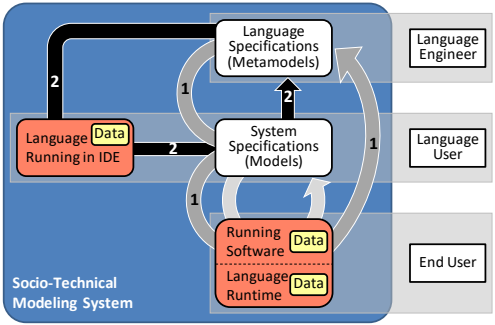
1. Free the language user from the implementation of :
   - The feedback loop
   - The trade-off analysis
2. Allow continuous and automatic evolution of itself

L-MODA | Languages, Models, and Data



L-MODA Conceptual Framework for Self-Adaptable Languages

L-MODA | Languages, Models, and Data

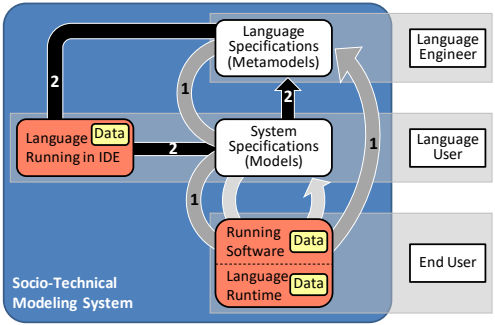


L-MODA Conceptual Framework for Self-Adaptable Languages

1) Runtime Feedback Loop

Use run-time data, model & metamodel
→ adaptation of language semantics

L-MODA | Languages, Models, and Data



L-MODA Conceptual Framework for Self-Adaptable Languages

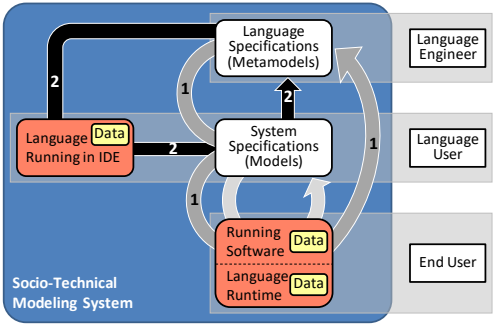
1) Runtime Feedback Loop

Use run-time data, model & metamodel
→ adaptation of language semantics

2) Design Feedback Loop

Use design-time data, models & metamodel
→ adaptation of syntax, pragmatics & semantics

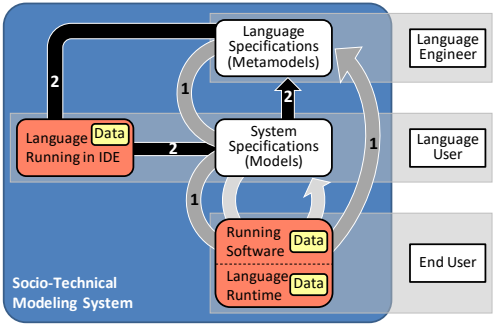
L-MODA | Stakeholders



L-MODA Conceptual Framework for Self-Adaptable Languages

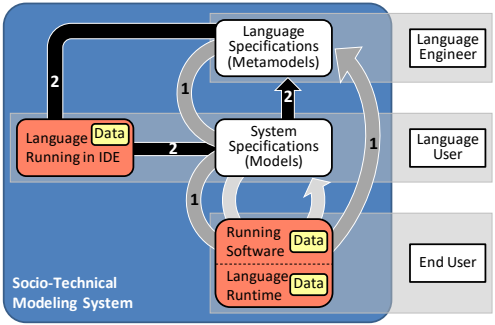
L-MODA | Stakeholders

Various uses of the feedback loops ...



L-MODA Conceptual Framework for Self-Adaptable Languages

L-MODA | Stakeholders



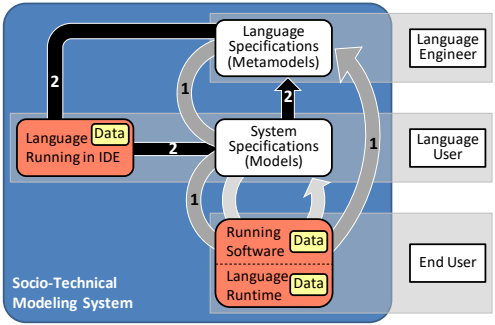
L-MODA Conceptual Framework for Self-Adaptable Languages

Various uses of the feedback loops ...

Examples for the Runtime Feedback Loop :

Delegation of responsabilities

L-MODA | Stakeholders



L-MODA Conceptual Framework for Self-Adaptable Languages

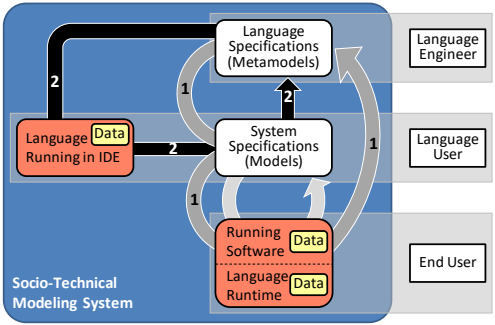
Various uses of the feedback loops ...

Examples for the Runtime Feedback Loop :

> Language engineer in complete control
> Tailor the language to a particular trade-off

Delegation of responsabilities

L-MODA | Stakeholders



L-MODA Conceptual Framework for Self-Adaptable Languages

Various uses of the feedback loops ...

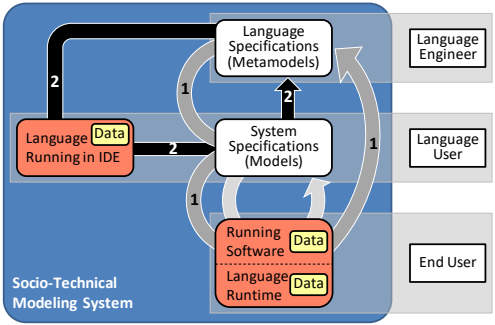
Examples for the Runtime Feedback Loop :

Language engineer in complete control
Tailor the language to a particular trade-off

Language user custom adaptations
Configure the adaptations for a system

Delegation of responsabilities

L-MODA | Stakeholders



L-MODA Conceptual Framework for Self-Adaptable Languages

Various uses of the feedback loops ...

Examples for the Runtime Feedback Loop :

Language engineer in complete control
Tailor the language to a particular trade-off

Language user custom adaptations
Configure the adaptations for a system

End-user preferences
Indicate preference for trade-offs

Delegation of responsabilities

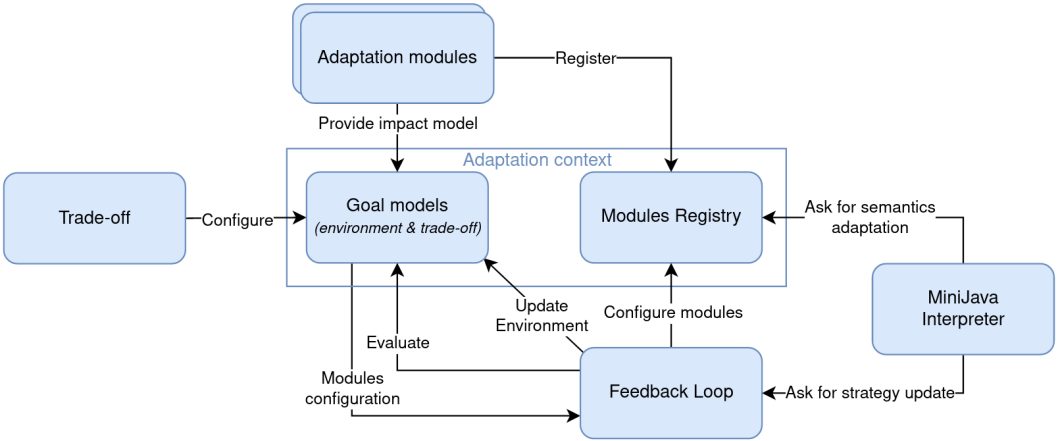
Experimentation

The case of Self-Adaptable Virtual Machines

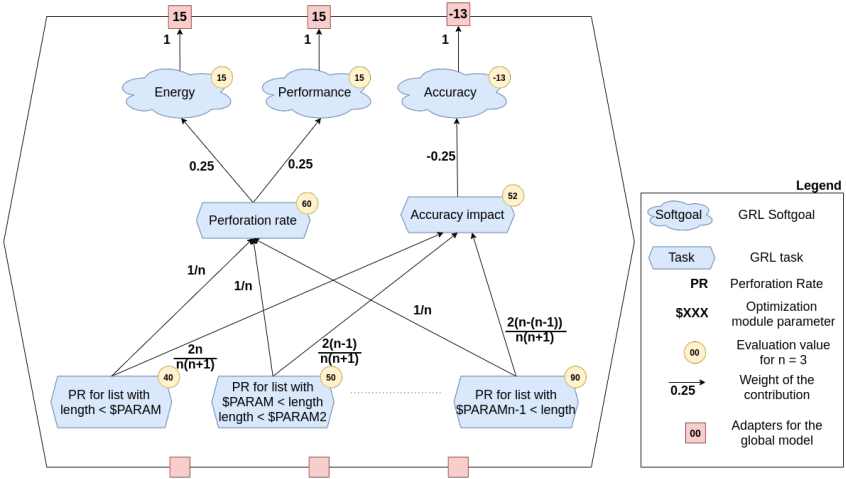
What are Self-Adaptable Virtual Machines

- A specific case of Self-Adaptable Languages
- Runtime Feedback loop in language operational semantics
- *In our experiment :* Pluggable architecture with delegation of responsibilities

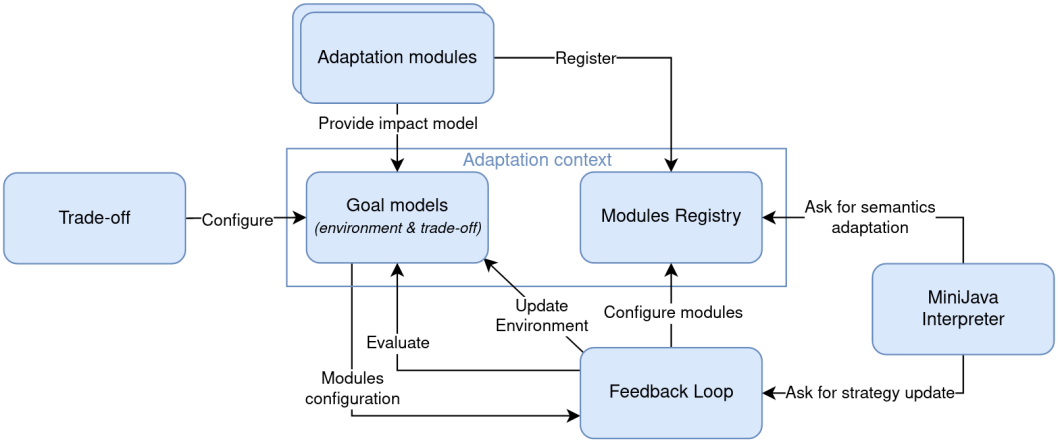
Architecture of Self-Adaptable Virtual Machines (MiniJava)



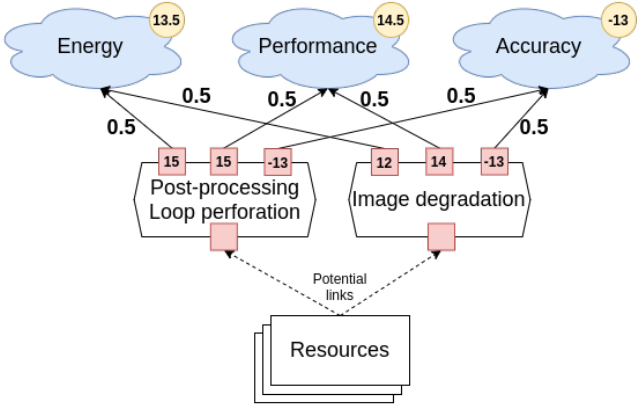
Impact Models

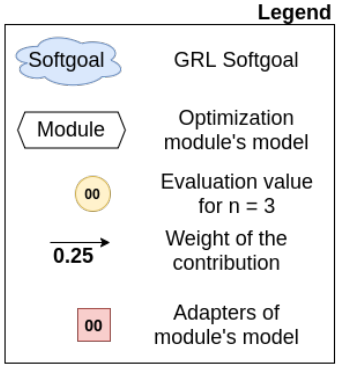


Architecture of Self-Adaptable Virtual Machines (MiniJava)

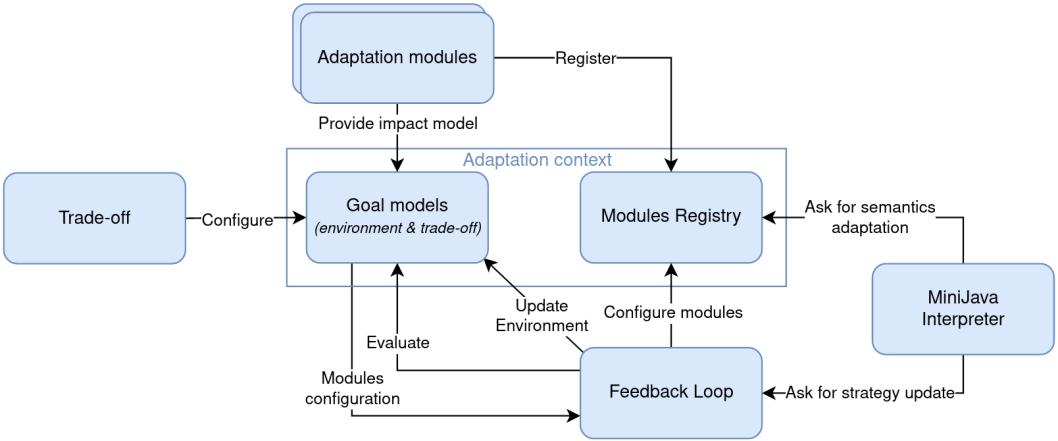


Global Goal Model

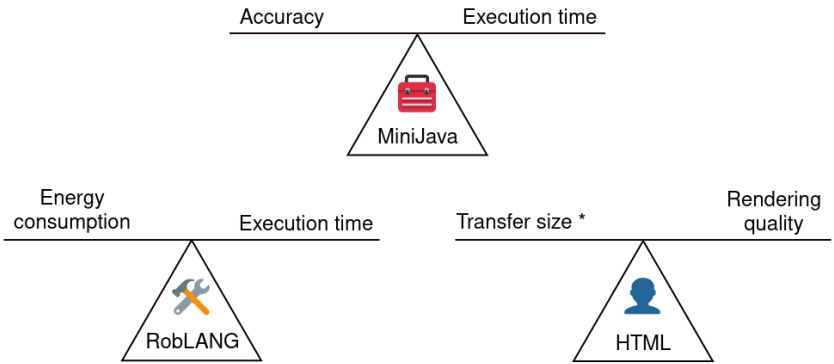




Architecture of Self-Adaptable Virtual Machines (MiniJava)



Motivating Examples



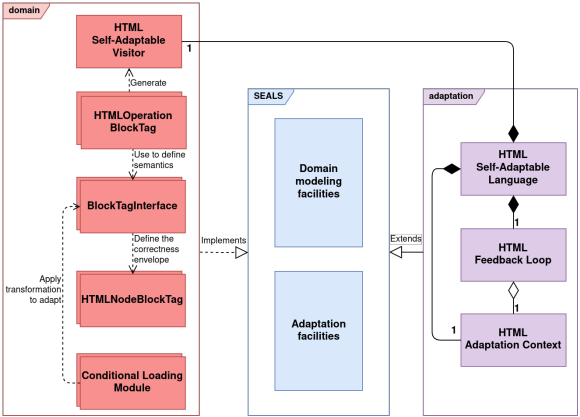
* Transfer size is proportional to energy consumption (Cf. `https://www.websitecarbon.com/`)

Evaluate the relevance of proposed adaptation

TL;DR : Good results but ...

- Correct adaptations of MiniJava
- Up to 10x more actions on RobLANG
- Energy reduction from -8.7% to 97.2% with a mean of 63.8% [54.2%, 73.4%]
- Performance overhead
- Lack of control on the adaptations
- Deal with the diversity of programs oblivious of the adaptations performed

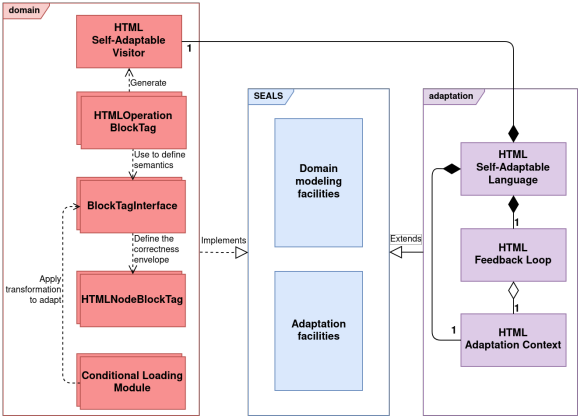
SEALS : A Framework for Building Self-Adaptable Virtual Machines



Approach overview on the HTML use case

SEALS : A Framework for Building Self-Adaptable Virtual Machines

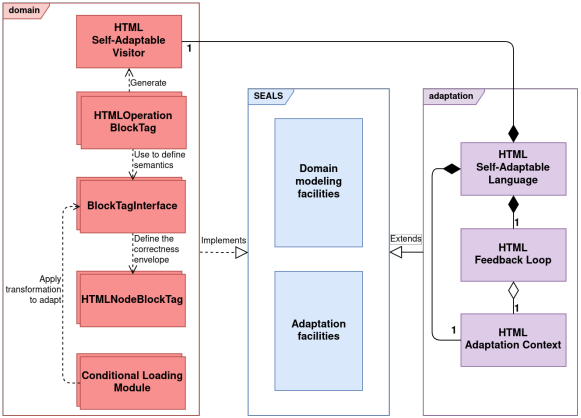
- Modeling of domain concepts
  1. Define the abstract syntax
  2. Create the correctness envelope
  3. Implement the operational semantics



Approach overview on the HTML use case

SEALS : A Framework for Building Self-Adaptable Virtual Machines

- Modeling of domain concepts
  1. Define the abstract syntax
  2. Create the correctness envelope
  3. Implement the operational semantics
- Adaptation process' specialization
  1. Specialize the Adaptation Context
  2. Specialize the Feedback loop
  3. Connect the components



Approach overview on the HTML use case

Adaptive Structural Operational Semantics

Join work with L. Thomas van Binsbergen and Damian Frölich

Adaptive Structural Operational Semantics

Based on a language definition :

Adaptive Structural Operational Semantics

Based on a language definition :

- Abstract syntax as metamodel
- Semantic domain merged in the metamodel
- Modular definition of the semantics (I-MSOS)

Adaptive Structural Operational Semantics

Based on a language definition :

- Abstract syntax as metamodel
- Semantic domain merged in the metamodel
- Modular definition of the semantics (I-MSOS)

To make it adaptive we need :

Adaptive Structural Operational Semantics

Based on a language definition :

- Abstract syntax as metamodel
- Semantic domain merged in the metamodel
- Modular definition of the semantics (I-MSOS)

To make it adaptive we need :

- Additional semantics rules for adaptation
- Mechanism for adaptation rule introduction
- Dynamic selection of semantics rule to apply

Meta-language for Original Semantics

```
1  model imp.ecore with sd
2
3  rule program,
4      Program(command) -> Program(newcommand)
5  resolve
6      command -> newcommand
7
8  rule program_error,
9      Program(command) -> sd.RuntimeError()
10 resolve
11     command -> termination sd.RuntimeError()
12
13 rule assign_set,
14     Assignment(name, sd.Integer(n))
15     ->
16     sd.NilValue()
17 bind
18     self.value = sd.Integer(n)
```

```
1  rule if_then,
2      If(sd.Boolean(b), c1, c2) -> c1
3  where
4    b == true
```

- Model merging metamodel and dynamic information

Meta-language for Original Semantics

```
1  model imp.ecore with sd
2
3  rule program,
4      Program(command) -> Program(newcommand)
5  resolve
6      command -> newcommand
7
8  rule program_error,
9      Program(command) -> sd.RuntimeError()
10 resolve
11     command -> termination sd.RuntimeError()
12
13 rule assign_set,
14     Assignment(name, sd.Integer(n))
15     ->
16     sd.NilValue()
17 bind
18     self.value = sd.Integer(n)
```

```
1  rule if_then,
2      If(sd.Boolean(b), c1, c2) -> c1
3  where
4    b == true
```

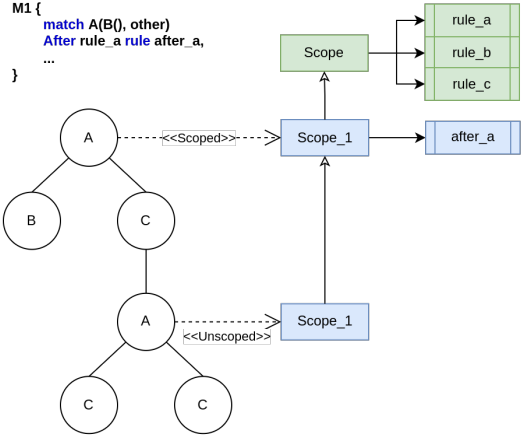
- Model merging metamodel and dynamic information
- A set of semantic rules
  - Conclusion as reduction over concepts
  - Reduction premises
  - Side condition
  - Binding computed values

Meta-language for Semantics Adaptation

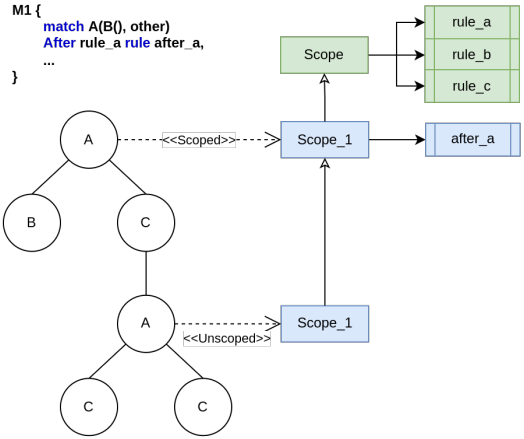
```
1  model imp.ecore with sd
2  semantics imp.sem
3
4  ApproximateDouble{
5
6      match Assignement(VarRef(def), expr)
7      where def.type == Float
8
9      Before binop_rhs rule binop_rhs_f,
10         Binop(Double(n1), a2)
11         ->
12         Binop(Float(n1), a2)
13
14     Before binop_result rule binop_result_f,
15         Binop(Number(n1), Double(n2))
16         ->
17         Binop(Number(n1), Float(n2))
18 }
```

- Dependence to the semantics
- Pointcut definition
  - Structural matching
  - Additional constraints
- Adaptation rules
  - Kind of adaptation rule
  - Affected rule in semantics
  - Adaptation semantic rule

Scopes of semantics rules

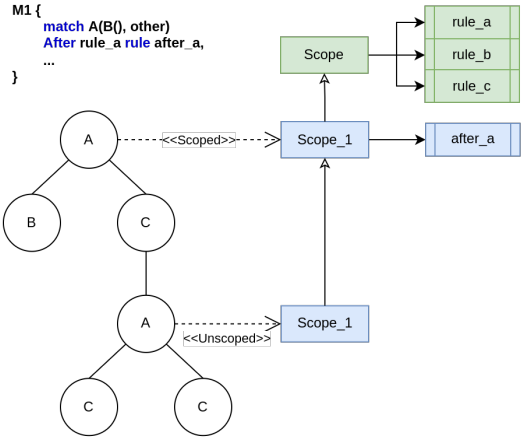


Scopes of semantics rules



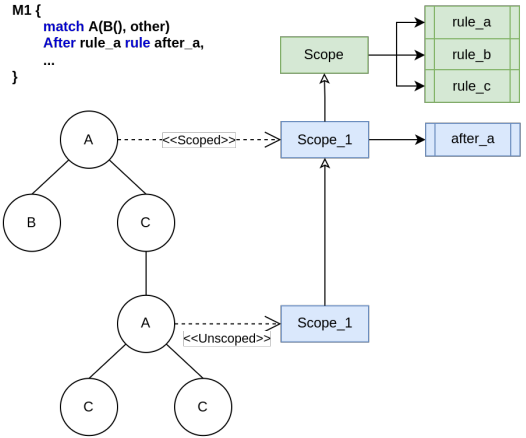
- Original semantics rules defined in the global scope

Scopes of semantics rules



- Original semantics rules defined in the global scope
- Adaptation introduce new rules through new scopes

Scopes of semantics rules



- Original semantics rules defined in the global scope
- Adaptation introduce new rules through new scopes
- Instantiation of scopes defined using match expression

ASOS as family of Transition systems

An adaptive language is a structure $\langle \Gamma, A, R, T, \psi \rangle$, such that :

$\forall \rightarrow \in R, \langle \Gamma, A, \rightarrow, T_{\rightarrow} \subseteq T \rangle$ is a TTS.

$\psi \colon P \times \Gamma \rightarrow \Gamma \times R$, selects the currently active transition system based on the active environment and the current state.

Still ongoing

- Try to prove we can generate deterministic semantics under some assumptions
- Finish the code generator based on the meta-language
- Do the experimentation and write the paper

Articles on the subject

- The concept of Self-Adaptable Language and its conceptual framework
  G. Jouneaux, O. Barais, B. Combemale, *et al.*, "Towards Self-Adaptable Languages," in *Onward! 2021*, Chicago, United States, Oct. 2021. [Online]. Available: `https://hal.inria.fr/hal-03318816`
- A framework to implement Self-Adaptable Virtual Machines
  G. Jouneaux, O. Barais, B. Combemale, *et al.*, "SEALS: A framework for building Self-Adaptive Virtual Machines," in *SLE 2021*, Chicago, United States, Oct. 2021. DOI: `10.1145/3486608.3486912`
- Ongoing work on specification of adaptive semantics
  Planned submission to PLDI 2023 (deadline November, 10th)

Perspectives

- Test and debug programs running on SALs and adaptations
- Specification of Self-Adaptive Operational Semantics feedback loops
- Writing my PhD thesis

Thanks for your attention !