Towards Self-Adaptable Languages

Gwendal Jouneaux Olivier Barais
Benoit Combemale Gunter Mussbacher

UNIVERSITÉ DE RENNES 1 — UMR IRISA — Inria — McGill University

Context

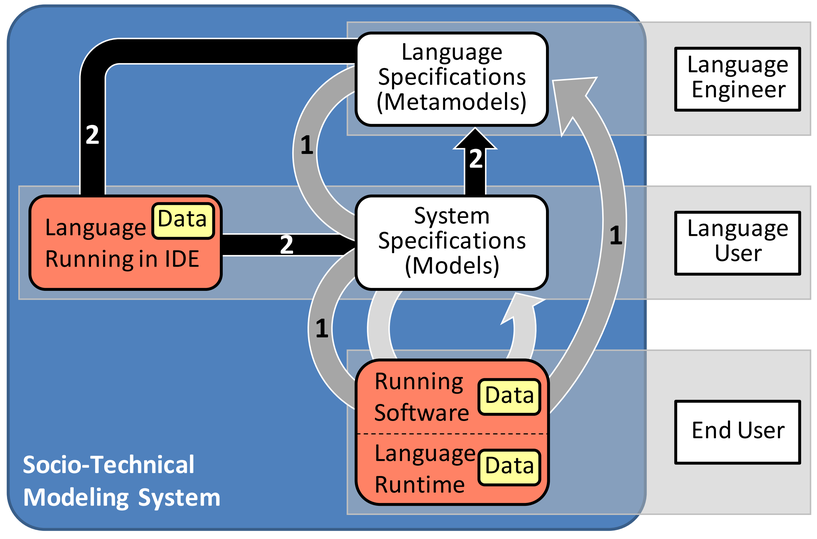
Nowadays software evolve in complex and changing environment (*e.g.*, Cloud) and need dynamic adaptation to best deliver the service (*e.g.*, Netflix)

On the other hand, Software languages can abstract concerns into high level constructs (*e.g.*, memory management)

Vision : abstract self-adaption into high level language constructs

This abstraction is not for when self-adaptation is the primary concern, such as autonomous cars, but when its a secondary but nevertheless an important concern that the developer don't want/need to manage manually.

Self-Adaptable Languages



L-MODA Reference 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

Configuration can be delegated to other stakeholders
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

Research Road-map

Support of the Runtime Feedback Loop

- Feedback loop configuration
- A reference framework for common implementation [1]
- ...

Support of the Design Feedback Loop

- Model the development context
- Detect evolution opportunities
- Navigate in evolution of programs
- ...

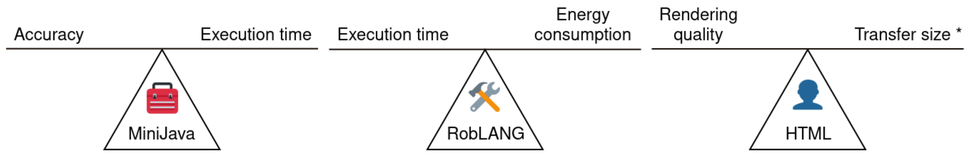
For more details take a look at our paper's research road-map

Self-Adaptable Virtual Machines (Experimentations)

What are Self-Adaptable Virtual Machines ?

" *Reification of the Runtime Feedback loop in language operational semantics* "

Motivating Examples



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

Adaptations proposed

MiniJava
Applied Approximate Loop Unrolling on the Sobel filter algorithm

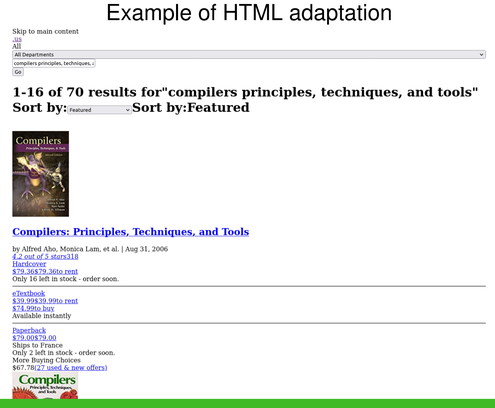
RobLANG
Applied a motor speed reduction on basic actions (Turn, Move, Squares)

HTML

- Conditional loading of resources
- Perforation of HTML lists
- Image degradation

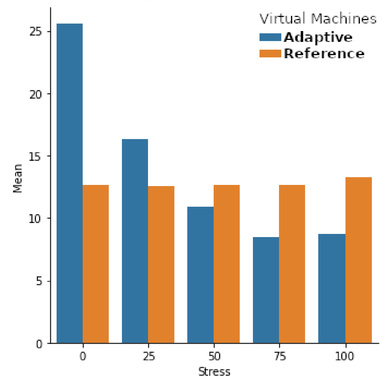
→ Applied on the top 100 websites

Example of HTML adaptation

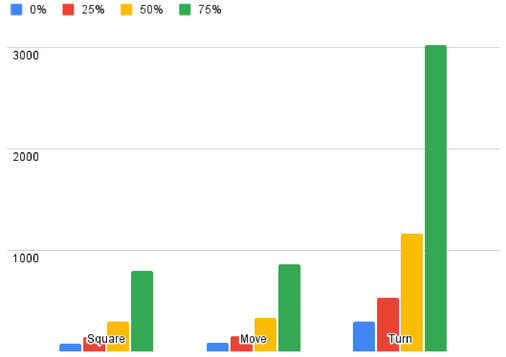


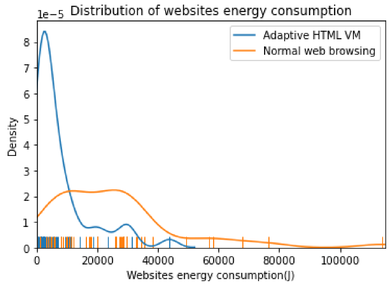
Results

Sobel filter mean execution time depending on the CPU usage



Number of actions performed by robot depending on the program and the percentage of time constraint relaxation





VMs	Trade-off reasoning	Feedback loop calls	Interaction with the domain
MiniJava	=	+	+
RobLang	+ +	+	-
HTML(JS)	=	=	-

Comparison of implementation simplicity
(+ in favor of language-level)

Discussion

Avoid difficulty of manual implementation
→ HTML/JS is a special case DSL/GPL

Adapt correctly but ...

- Performance overhead
- Only 45 of 100 websites works with HTML aggressive adaptations
- Lack of control on the adaptations

Take Away

- *Self-Adaptable Languages* **abstracts** the design and execution of *feedback loops*
- SALS encompass both the **design-time** environment and the **run-time** environment
- **Promising results** for *adaptations* of language **operational semantics**

References

[1] G. Jouneaux, O. Barais, B. Combemale, and G. Mussbacher, "SEALS: A Framework for Building Self-Adaptive Virtual Machines," in *SLE '21*, Oct. 2021.

DiverSE Diversity-Centric Software Engineering

G. Jouneaux, O. Barais, B. Combemale, and G. Mussbacher, "Towards Self-Adaptable Languages," in Onward! 2021.

Available on HAL : `https://hal.inria.fr/hal-03318816`