IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

ELASCRIPT: A DSL FOR CODING ELASTICITY IN CLOUD COMPUTING

Simon Dupont (*)
Salma Bouri, Frederico Alvares, Thomas Ledoux (**)

(*) Sigma Informatique
(**) IMT Atlantique campus Nantes, LS2N, INRIA

OUTLINE

1. PROBLEM

1.1 Cloud Elasticity

1.2 ElaStuff Overview

2. ELASCRIPT

2.1 Overview

2.2 Examples

2.3 Guarantees

3. VALIDATION

3.1 Testbed

3.2 Results and discussion

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

CLOUD ELASTICITY

Overview and Shortcomings

Overview

Adjust resources automatically according to the demand so as to satisfy a certain level of Quality of Service (QoS) while minimizing infrastructure costs

Mainly based on IaaS elasticity (VM scaling)

Shortcomings

Some limits in terms of responsiveness (e.g., VM startup time)

Risks in terms of

- ✓ Over-provisioning: highly cost
- ✓ Under-provisioning: SLA violation

Why?

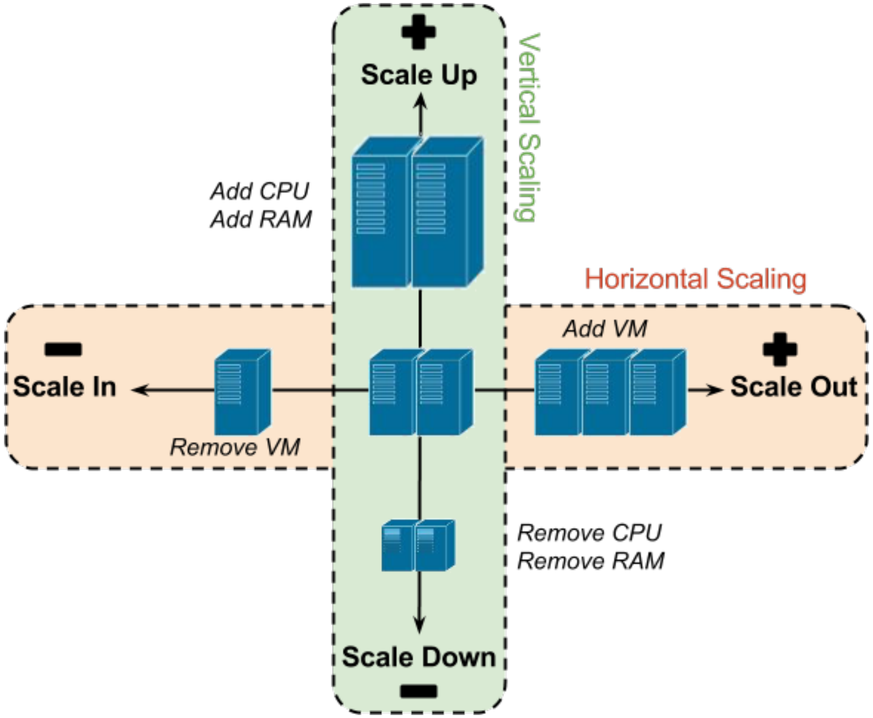
A problem of reactivity

A problem of resource granularity

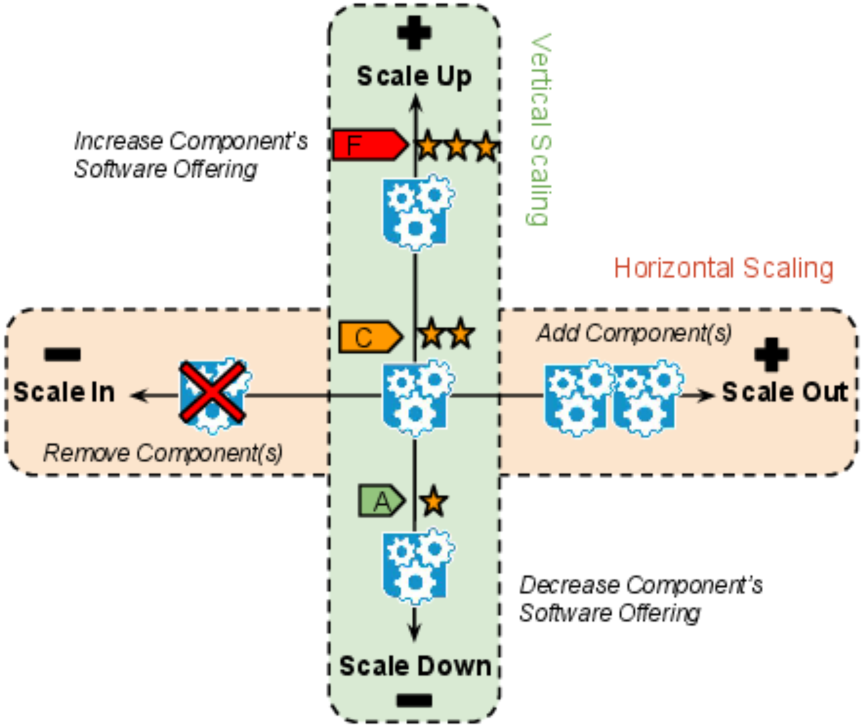
CLOUD ELASTICITY

Cross-layer elasticity

Objective: software layer (SaaS) may take part in the global elasticity process to cope with IaaS elasticity shortcomings



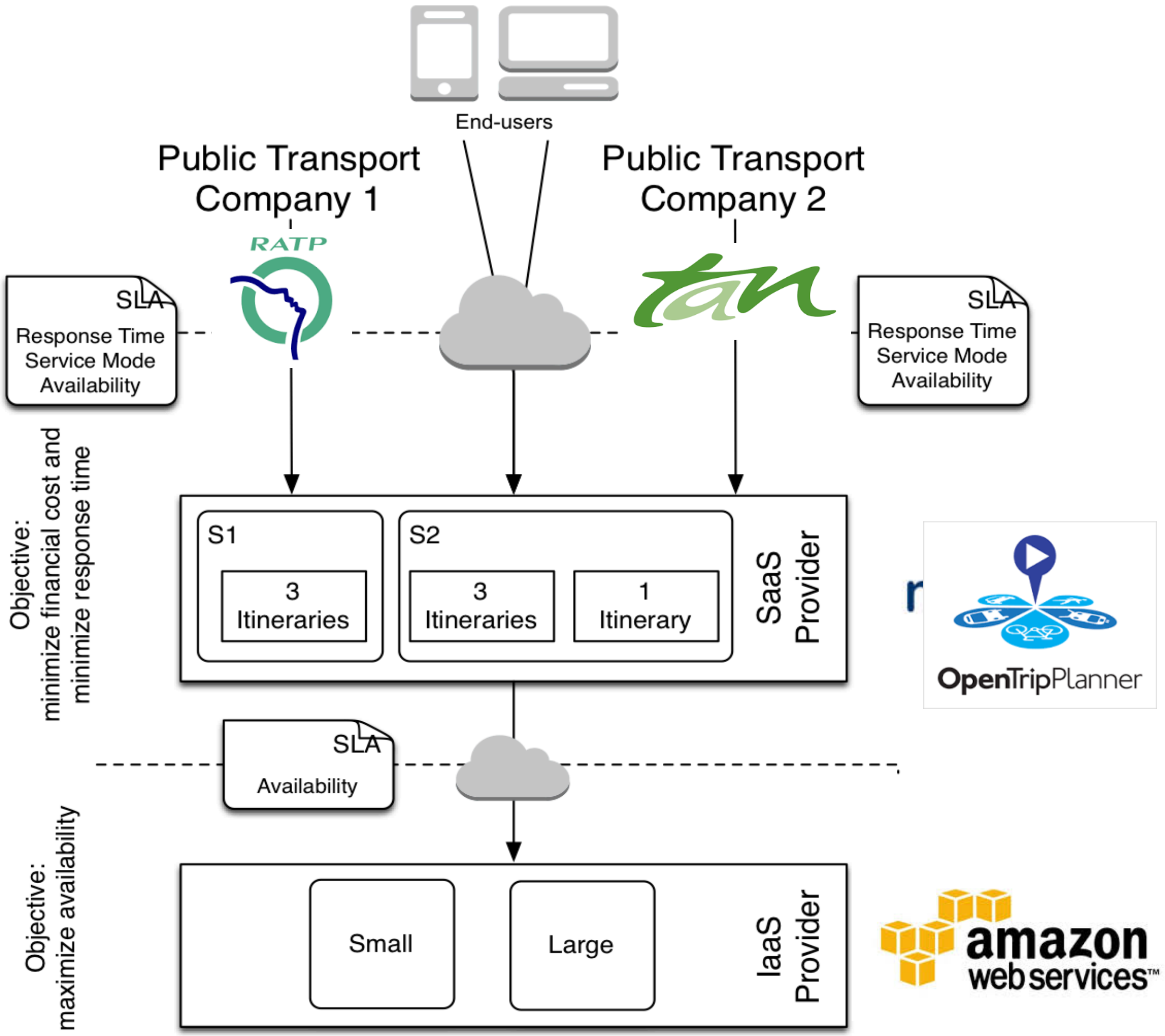
Elasticity @ IaaS Level



Elasticity @ SaaS Level

CROSS-LAYER ELASTICITY

Illustration



CROSS-LAYER ELASTICITY

Conclusion

Advantages

A finer scaling granularity

Fulfill SLA contracts (more reactivity)

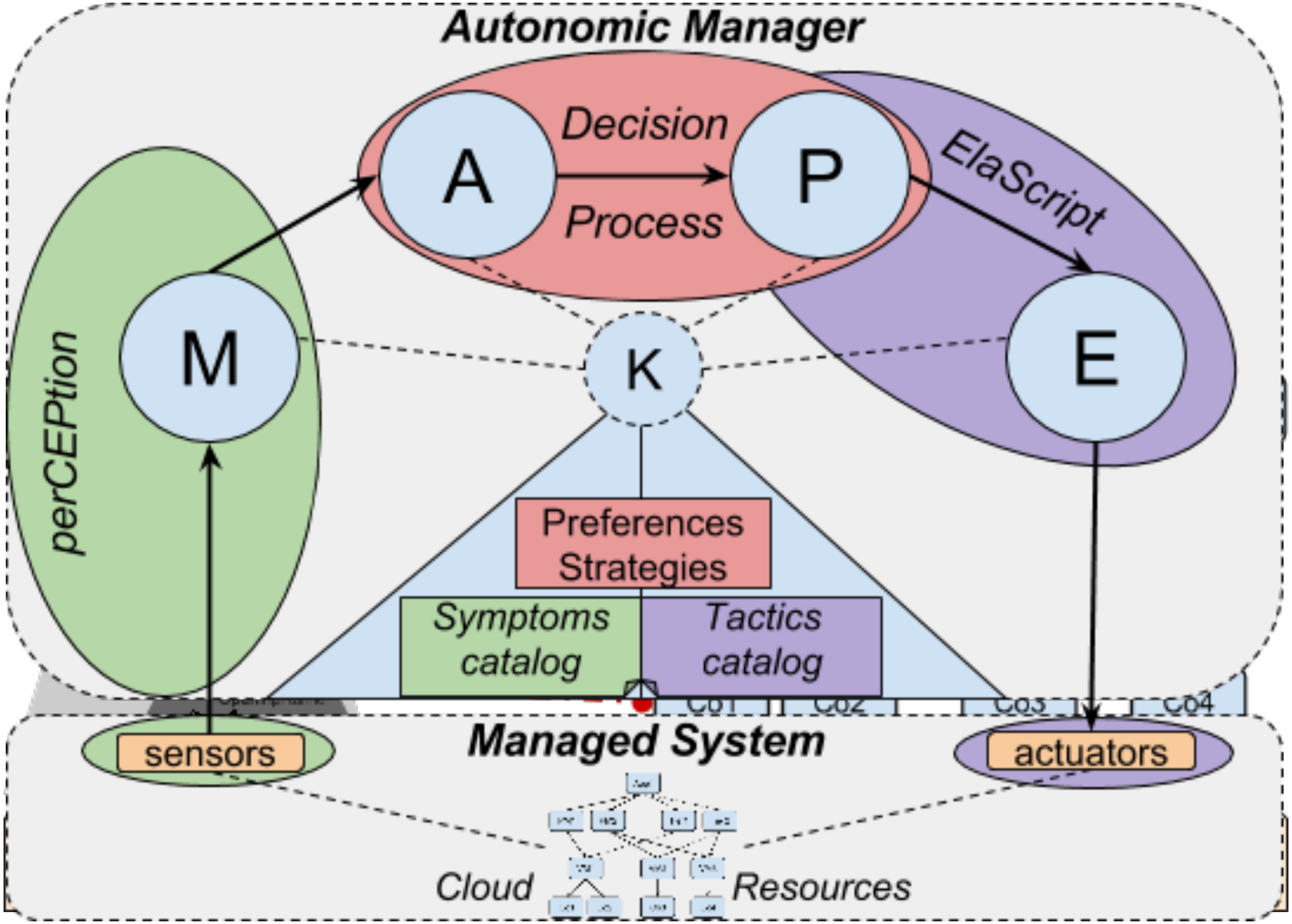
Minimize resources (economic and ecological pros)

Consequence

Cloud administrators can create more sophisticated reconfiguration plans and strategies (e.g., minimize energy consumption)

ELASTUFF

Overview



ELASCRIPT

Overview

Goal

A support for Cloud administrators to simply and safely express reconfiguration plans

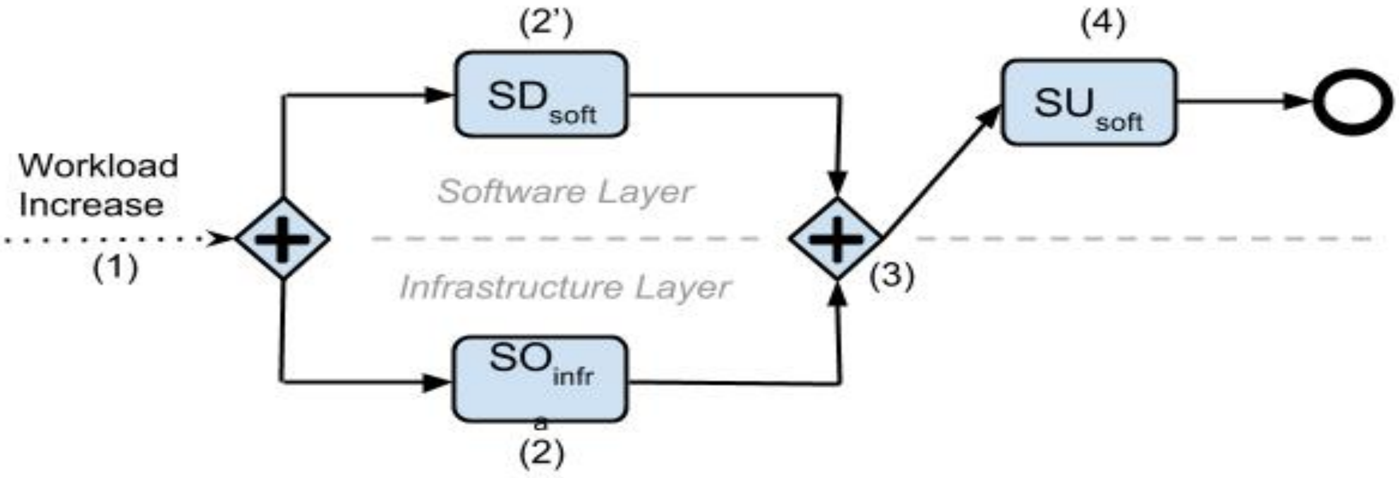
How ?

Manipulate resource graphs via 8 primitive APIs

- ✓ (Horizontal, vertical scaling) X (IaaS, SaaS)

Use a DSL (Domain Specific Language) to wire and update the graph

- ✓ Possibilities for static analysis, verification or optimization
- ✓ DSL constructs for coordination of cross-layer elasticity



ELASCRIPT

In a nutshell

A scripting language

Imperative, Java-like syntax, static typing, no new variables, only one kind of iterator

Guarantees

Naming, scoping, typing, business rules

Technologies used

EMF, Xtext, Xtend, Java framework Executor

tactic

```
begin
  when SymptomName (Type1 resource1 , . . . , TypeN resourceN)
  do
          resource1.actionX;
          . . .
          resourceN.actionY;
end
```

ELASCRIPT

Examples

Simple example

```
begin
  when TierHighRT (Tier myTier)
  do myTier.soi ( ”medium”);
end
```

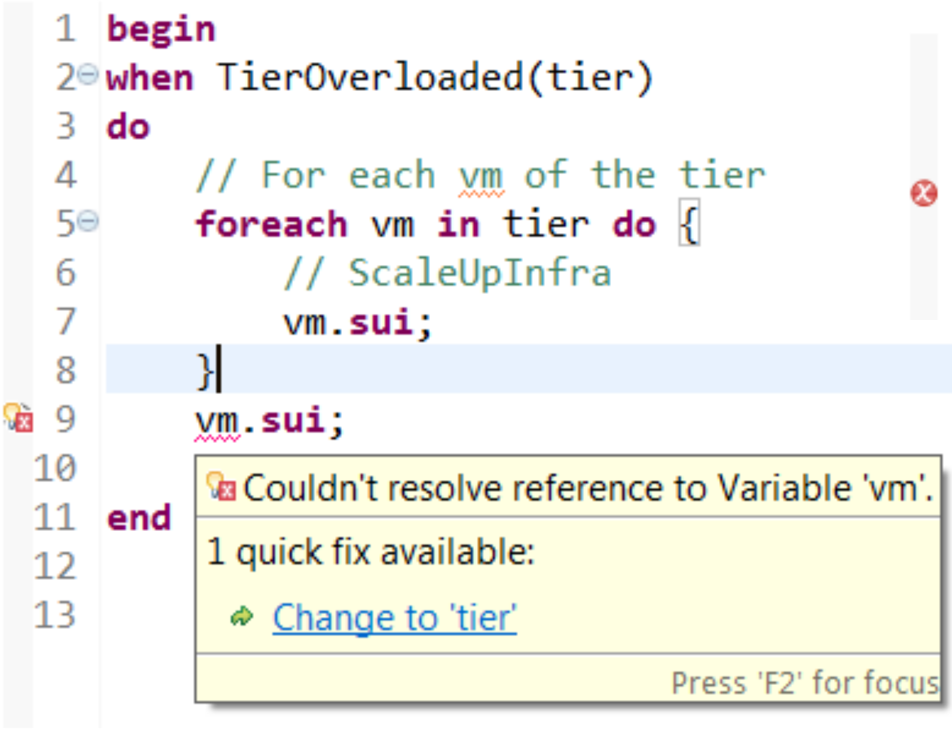
Cross-layer example

```
begin
  when Tier VMsOverloaded (Tier tier, List<VM> vms)
do
 [
   tier.soi (2 , ”small ”) ;
 | |
   foreach vm in vms do { vm.sds;}
 ]
 foreach vm in vms do {vm.sus; }
end
```

ELASCRIPT

Guarantees in Eclipse IDE

Scoping rules



```
begin
when TierOverloaded(tier)
do
    // For each vm of the tier
    foreach vm in tier do {
        // ScaleUpInfra
        vm.sui;
    }
    vm.sui;

end
```

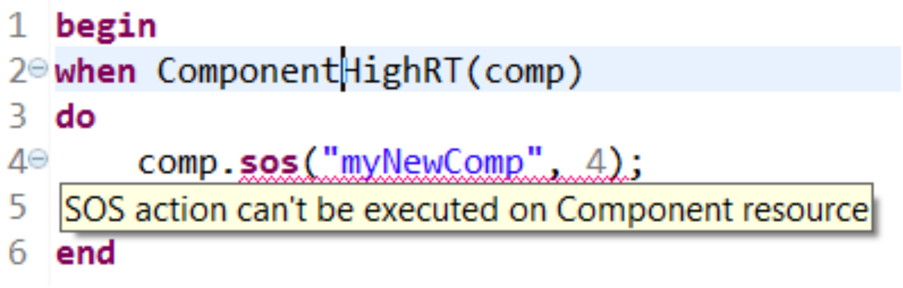
Couldn't resolve reference to Variable 'vm'.

1 quick fix available:

Change to 'tier'

Press 'F2' for focus

Typing rules



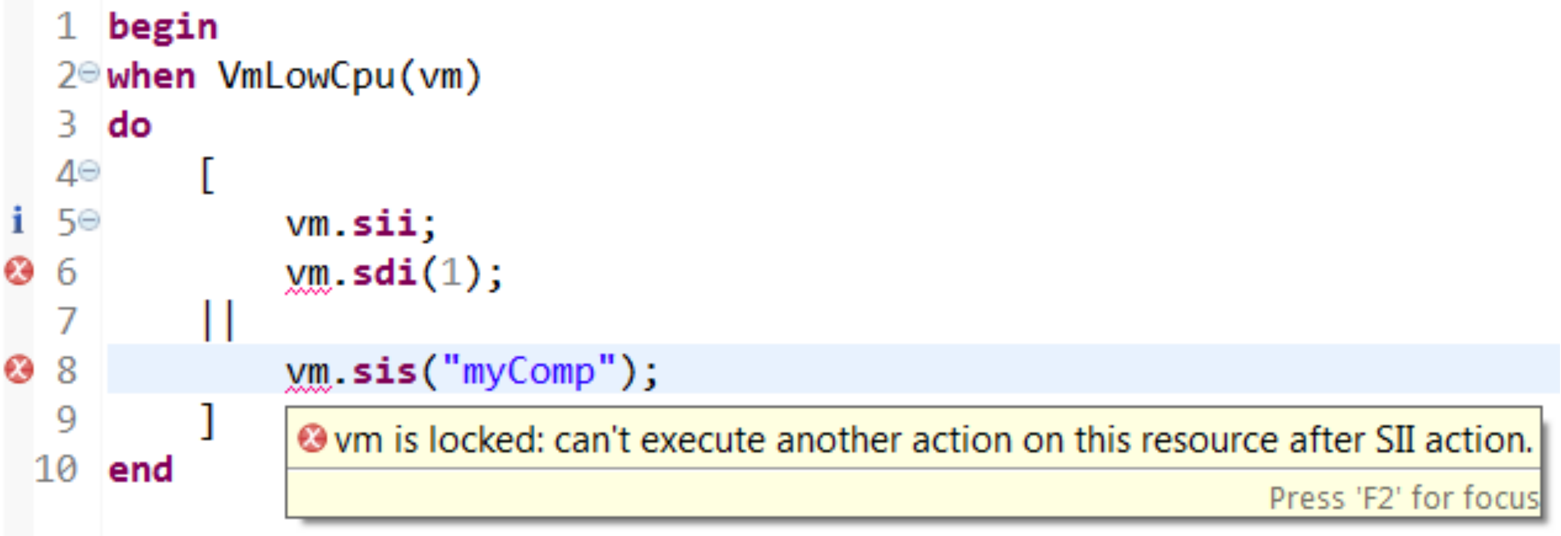
```
begin
when ComponentHighRT(comp)
do
    comp.sos("myNewComp", 4);

end
```

SOS action can't be executed on Component resource

ELASCRIPT

Guarantees in Eclipse IDE

Business rules



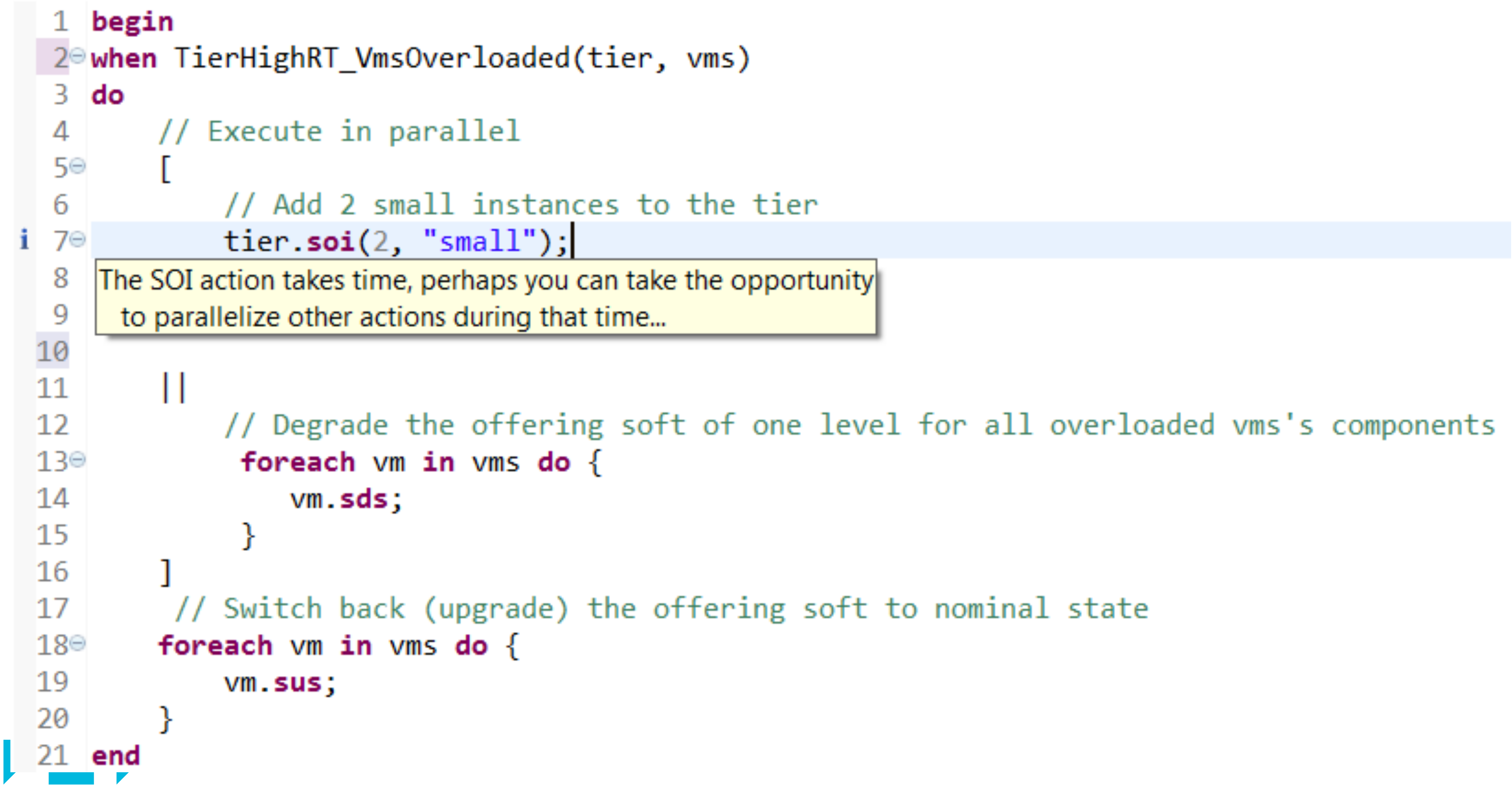
```
begin
when VmLowCpu(vm)
do
    [
        vm.sii;
        vm.sdi(1);
    ||
        vm.sis("myComp");
    ]
end
```

vm is locked: can't execute another action on this resource after SII action.

ELASCRIPT

Guarantees in Eclipse IDE

Business rules



```
begin
when TierHighRT_VmsOverloaded(tier, vms)
do
    // Execute in parallel
    [
        // Add 2 small instances to the tier
        tier.soi(2, "small");



    ||
        // Degrade the offering soft of one level for all overloaded vms's components
         foreach vm in vms do {
            vm.sds;
         }
    ]
     // Switch back (upgrade) the offering soft to nominal state
    foreach vm in vms do {
        vm.sus;
    }
end
```

The SOI action takes time, perhaps you can take the opportunity
to parallelize other actions during that time...

VALIDATION

Testbed

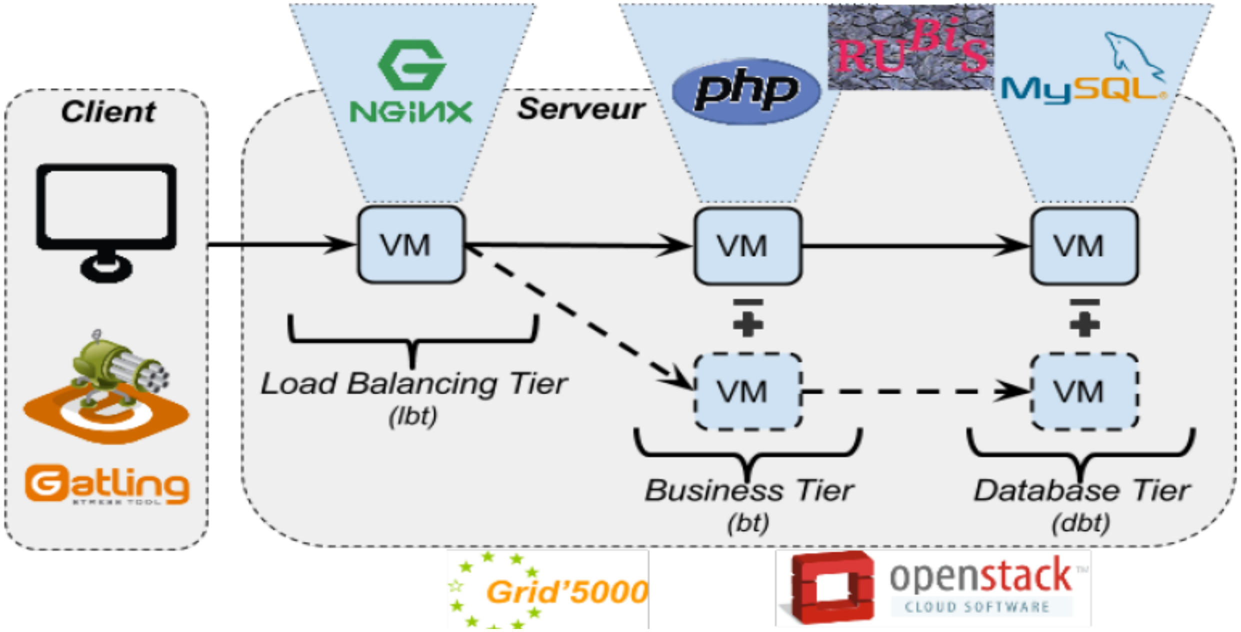
SaaS application

RUBiS (an eBay like auction site)

A three-tier architecture

Three Software offerings for recommendation:

- ✓ zero, one (user-to-user) or two (user-to-user and item-to-item)



VALIDATION

Testbed

IaaS infrastructure

OpenStack on Grid’5000 (large-scale testbed for experiment-driven research)

2 physical machines with wattmeter

Workload Traces

Real trace of FIFA'98

Elastuff Tactic

```
begin
  when App[Low | Normal | High]RT (App application)
  do application.sds( [2 | 1 | 0]);
end
```

Experiment Runs

Base line: a static version of RUBiS with 2 recommendations *vs* Elastuff one

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

VALIDATION

Results and discussion

QoS Criteria

Availability

- ✓ Baseline requests failed x330 more

Performance

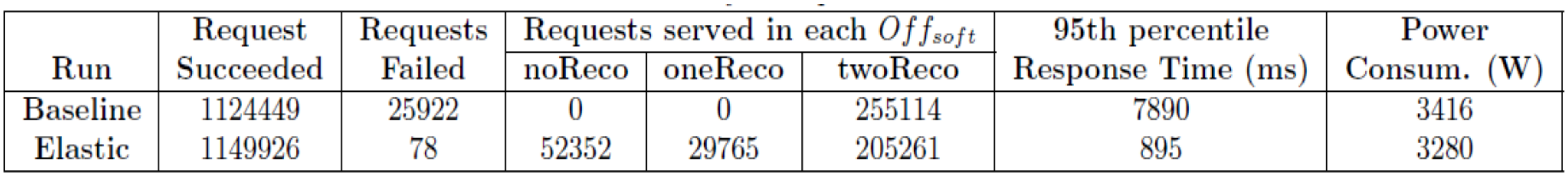
- ✓ Baseline response time x8

QoE

- ✓ Elastic version: 18% with 0 recommendation, 10% with only one

Energy consumption

- ✓ equivalent



	Request	Requests	Requests served in each Off_{soft}			95th percentile	Power
Run	Succeeded	Failed	noReco	oneReco	twoReco	Response Time (ms)	Consum. (W)
Baseline	1124449	25922	0	0	255114	7890	3416
Elastic	1149926	78	52352	29765	205261	895	3280

SLA contracts

Trade-off between criteria are indirectly managed by ElaScript

CONCLUSION

Conclusion

ElaScript: a DSL that enables Cloud administrators to simply and concisely program complex multi-layered reconfiguration plans

Perspectives

Consider new kinds of Cloud resources

- ✓ software containers (e.g., Docker), microservices

Consider new dimensions/actions

- ✓ VM migration

Renewable energy and Software elasticity

- ✓ Software offering should be relied on the presence of green resources
→ improve the carbon footprint and the software at the same time

Towards a *Elasticity-as-a-Service* offering

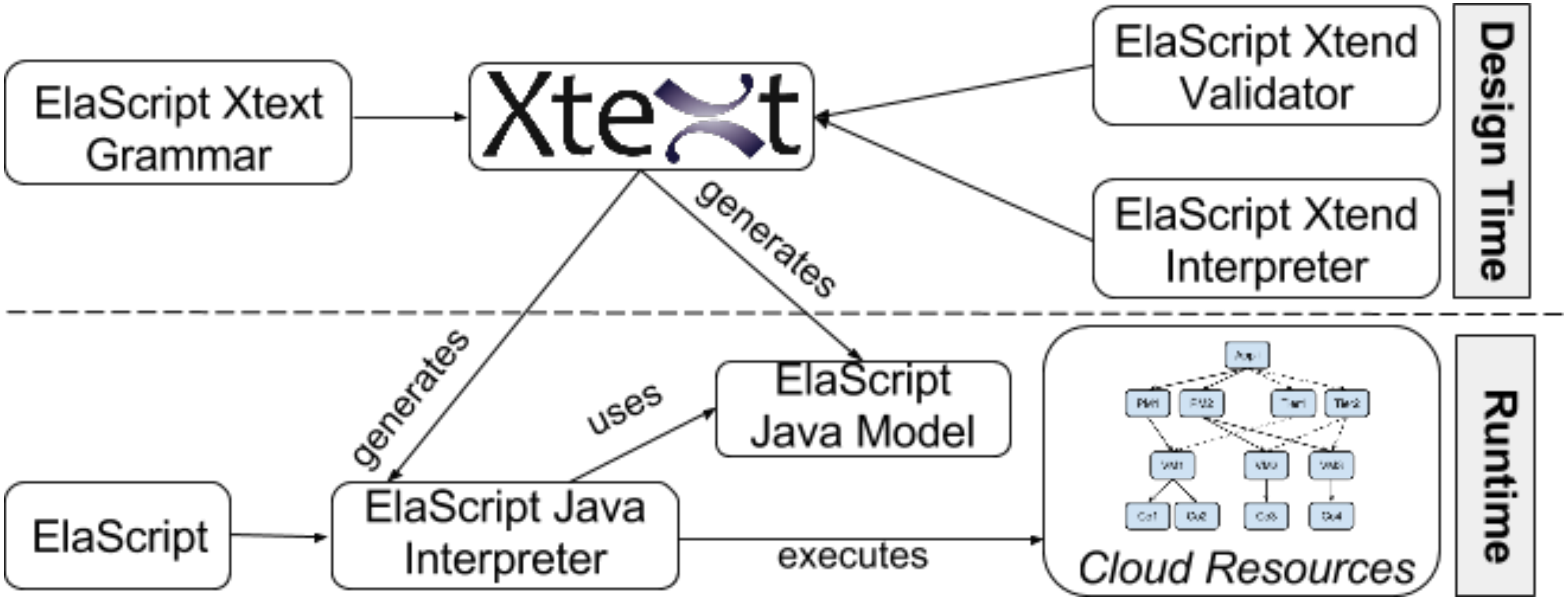
THANKS!

Any questions?

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

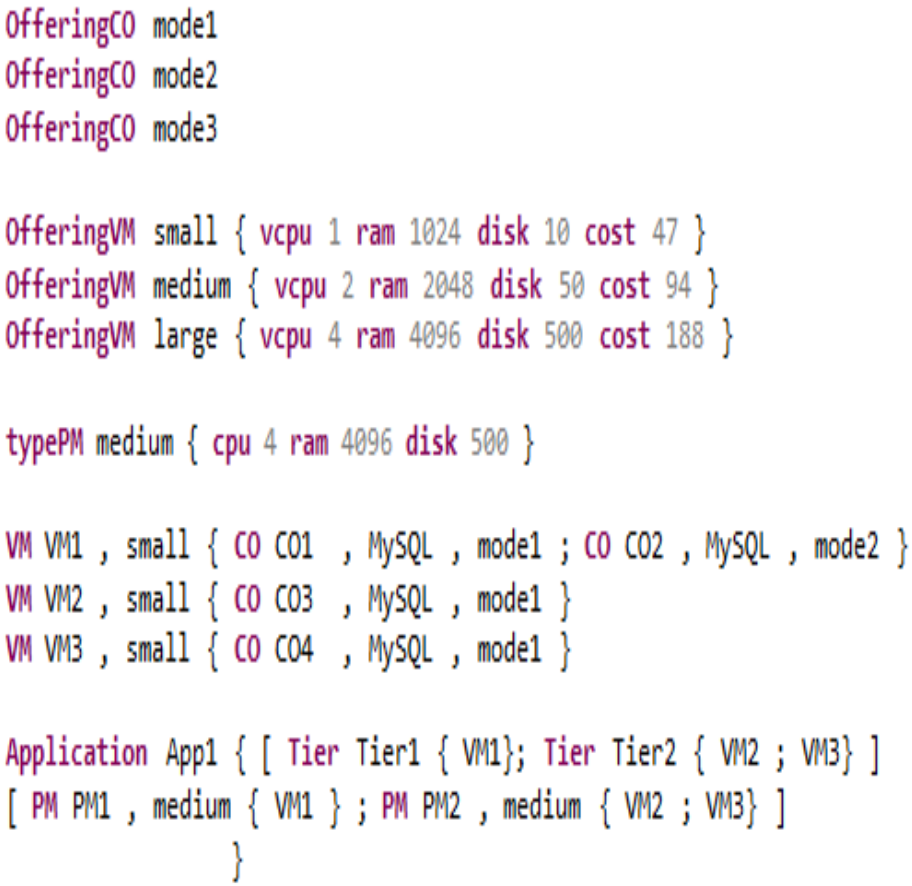
ELASCRIPT

Implementation

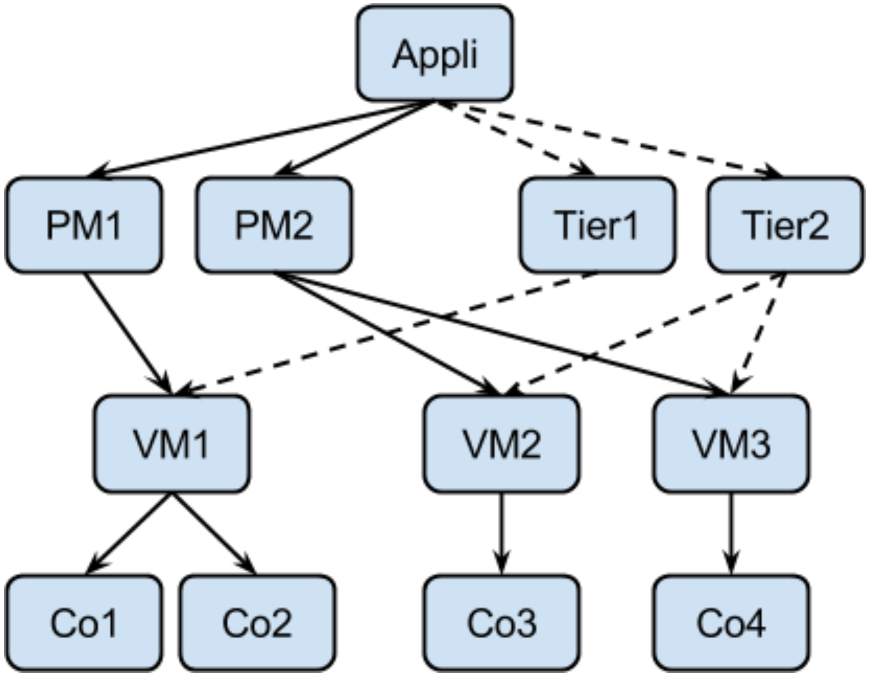


ELASTUFF ECOSYSTEM

How to define the resource graph?



```
OfferingCO mode1
OfferingCO mode2
OfferingCO mode3

OfferingVM small { vcpu 1 ram 1024 disk 10 cost 47 }
OfferingVM medium { vcpu 2 ram 2048 disk 50 cost 94 }
OfferingVM large { vcpu 4 ram 4096 disk 500 cost 188 }

typePM medium { cpu 4 ram 4096 disk 500 }

VM VM1 , small { CO CO1  , MySQL , mode1 ; CO CO2 , MySQL , mode2 }
VM VM2 , small { CO CO3  , MySQL , mode1 }
VM VM3 , small { CO CO4  , MySQL , mode1 }

Application App1 { [ Tier Tier1 { VM1}; Tier Tier2 { VM2 ; VM3} ]
[ PM PM1 , medium { VM1 } ; PM PM2 , medium { VM2 ; VM3} ]
                }
```


Appli
PM1
PM2
Tier1
Tier2
VM1
VM2
VM3
Co1
Co2
Co3
Co4