COMOT – Platform-as-a-Service for Software-defined Elastic Systems

Hong-Linh Truong

Distributed Systems Group
Vienna University of Technology
truong@dsg.tuwien.ac.at

http://dsg.tuwien.ac.at/research/viecom/

Acknowledgements

celar

PACIFIC CONTROLS

Joint work with:
Georgiana Copil, Schahram Dustdar, Alessio Gambi, Waldemar Hummer, Duc-Hung Le, Daniel Moldovan, Oliver Moser, Tien-Dung Nguyen

NOTE: The content includes some ongoing work

Outline

- Motivation
- Fundamental building blocks for software-defined elastic systems
  - Programming, deploying, controlling, monitoring and testing elasticity
- CoMoT architecture and its services
- Demo (next talk)
- Conclusions and future work

Recall - Elasticity in computing

1. Demand elasticity

   Elastic demands from consumers

2. Output elasticity

   Multiple outputs with different price and quality

3. Input elasticity

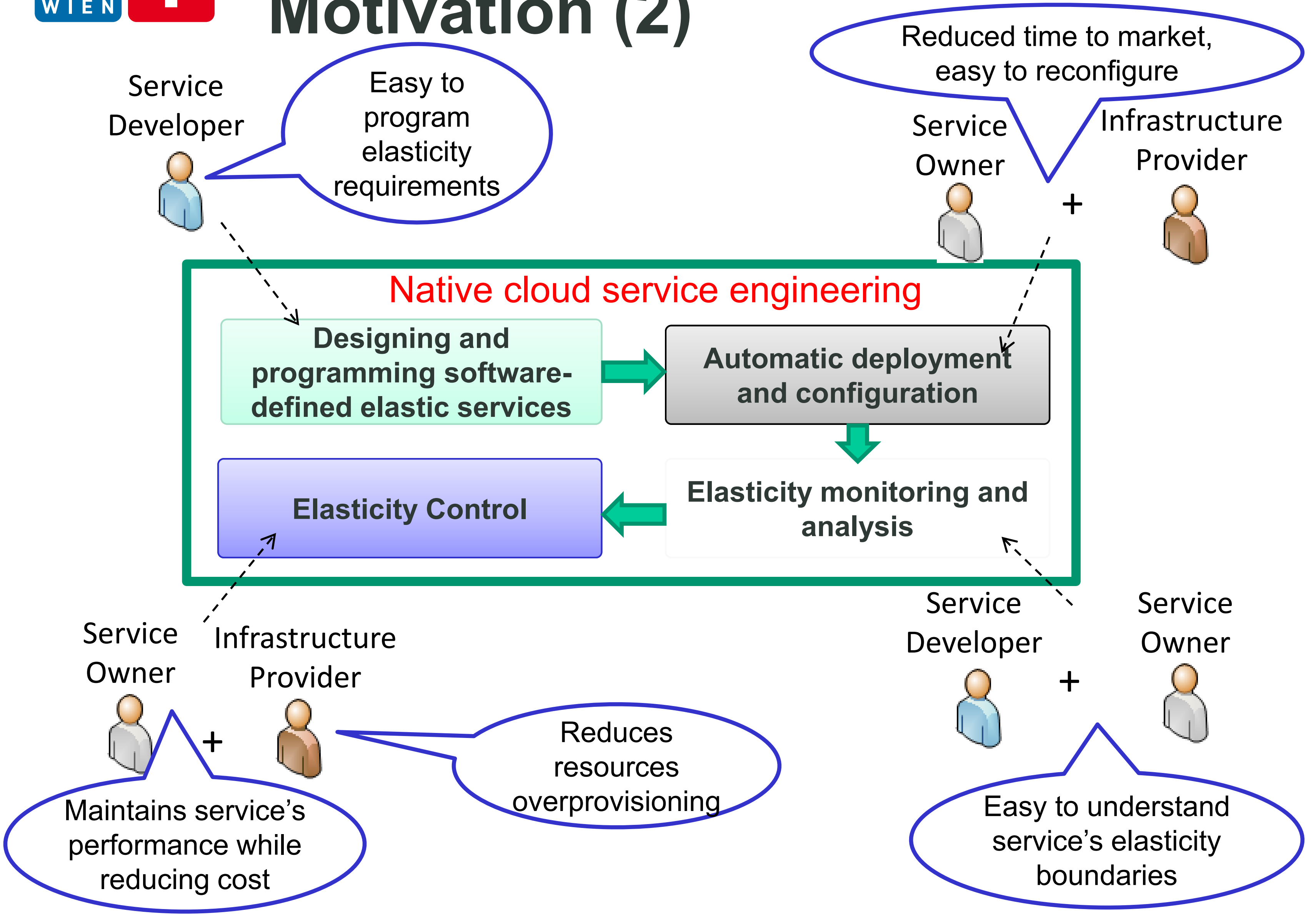
   Elastic data inputs, e.g., deal with opportunistic data

4. Elastic pricing and quality models associated resources

Motivation (1)

- Multi-dimensional elasticity is the fundamental requirement for native cloud services
  - resource elasticity, cost elasticity and quality elasticity
- But fragmented support on engineering elasticity requirements, execution, monitoring and testing, e.g.,
  - Only at resource elasticity at the IaaS level
  - Lack of elasticity monitoring for applications
  - Testing is not integrated with other phases
  - Elasticity within single clouds

Motivation (2)



Service Developer

Easy to program elasticity requirements

Reduced time to market, easy to reconfigure

Service Owner + Infrastructure Provider

Native cloud service engineering

Designing and programming software-defined elastic services

Automatic deployment and configuration

Elasticity monitoring and analysis

Elasticity Control

Service Owner + Infrastructure Provider

Maintains service's performance while reducing cost

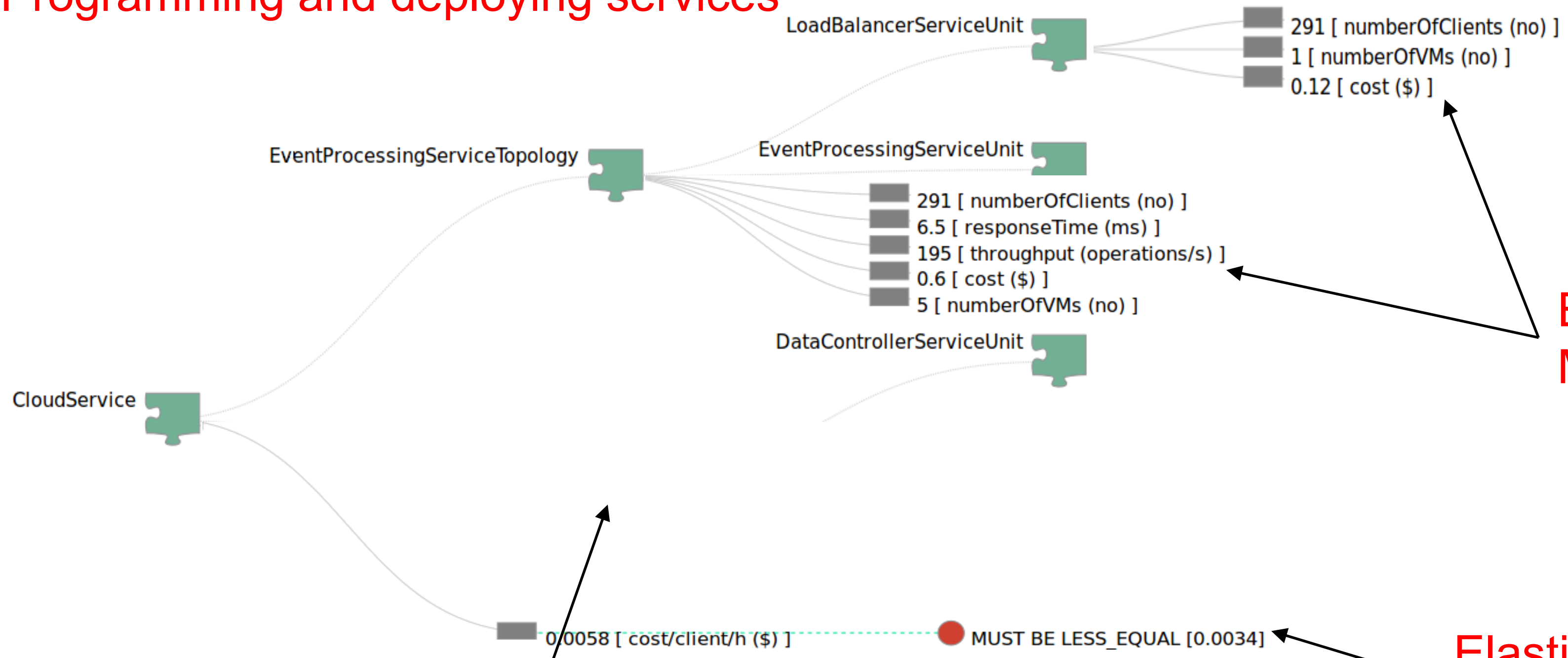
Reduces resources overprovisioning

Service Developer + Service Owner

Easy to understand service's elasticity boundaries

So what need to be done? A simple view

Programming and deploying services


LoadBalancerServiceUnit
291 [numberOfClients (no)]
1 [numberOfVMs (no)]
0.12 [cost ($)]
EventProcessingServiceTopology
EventProcessingServiceUnit
291 [numberOfClients (no)]
6.5 [responseTime (ms)]
195 [throughput (operations/s)]
0.6 [cost ($)]
5 [numberOfVMs (no)]
DataControllerServiceUnit
CloudService
0.0058 [cost/client/h ($)]
MUST BE LESS_EQUAL [0.0034]

Elasticity Metrics

Elasticity Requirements

Elasticity capabilities (e.g. scale IN/OUT)

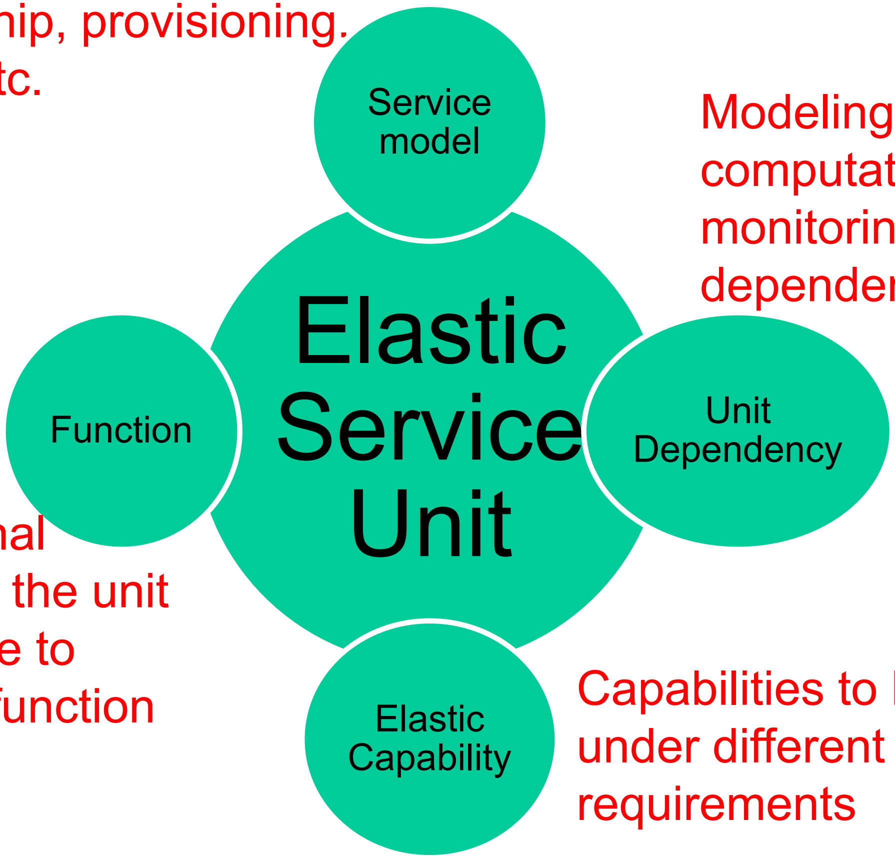
Fundamental building blocks for the elasticity

- Conceptualizing and modeling elastic objects and execution environments
  - So we can manage diverse types of artifacts and their runtime in a similar manner
- Defining and capturing elasticity primitive operations associated with elastic objects and environments
- Recommending and Programming elastic objects
  - a software-defined elastic system (SES) is built from elastic objects
- Runtime deploying, control, monitoring and testing techniques for elastic objects

Elastic service units as cloud programming objects

Consumption, ownership, provisioning, price, etc.

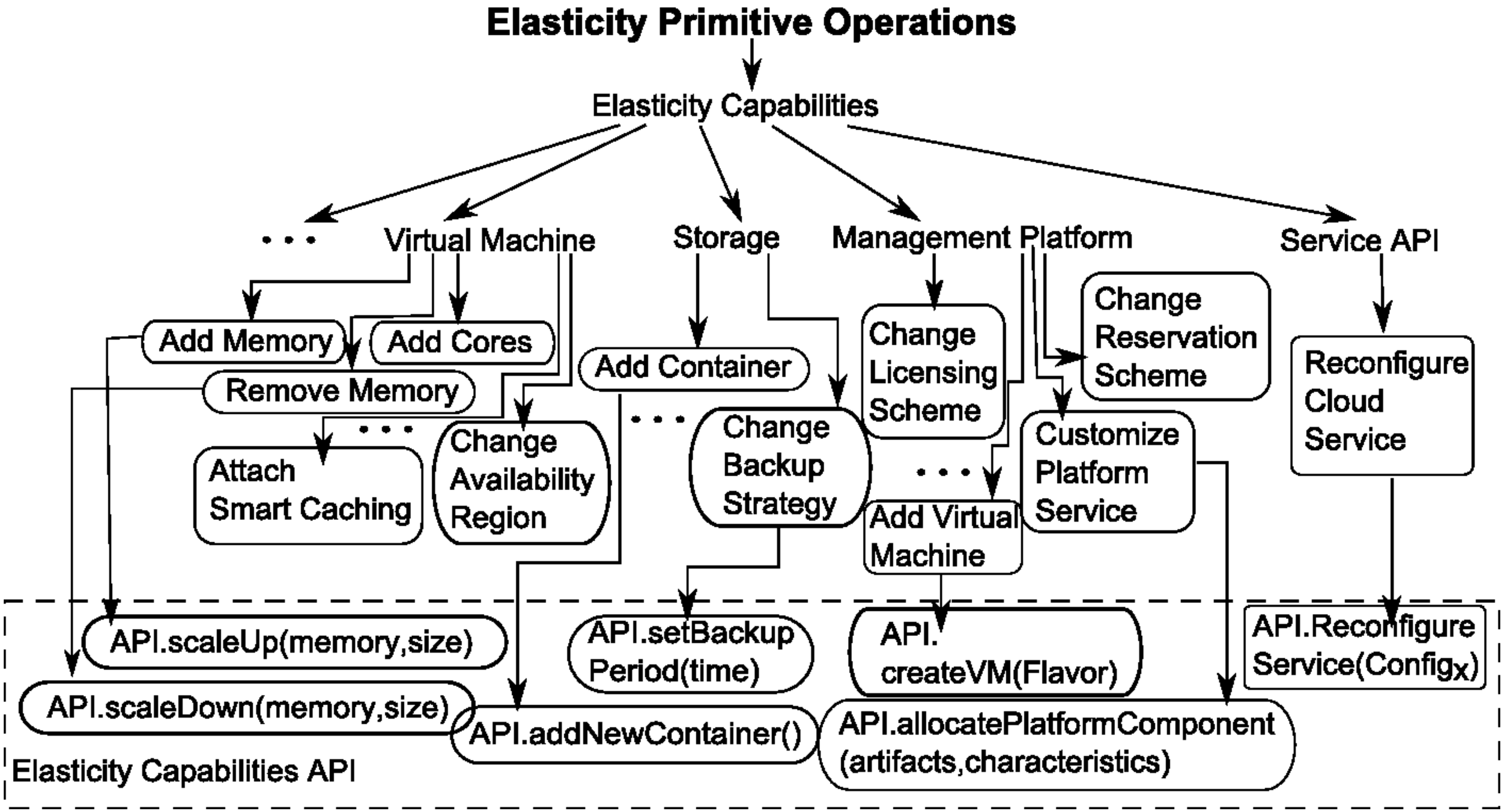
Modeling type of units (e.g., computation, data, monitoring,) and their dependencies



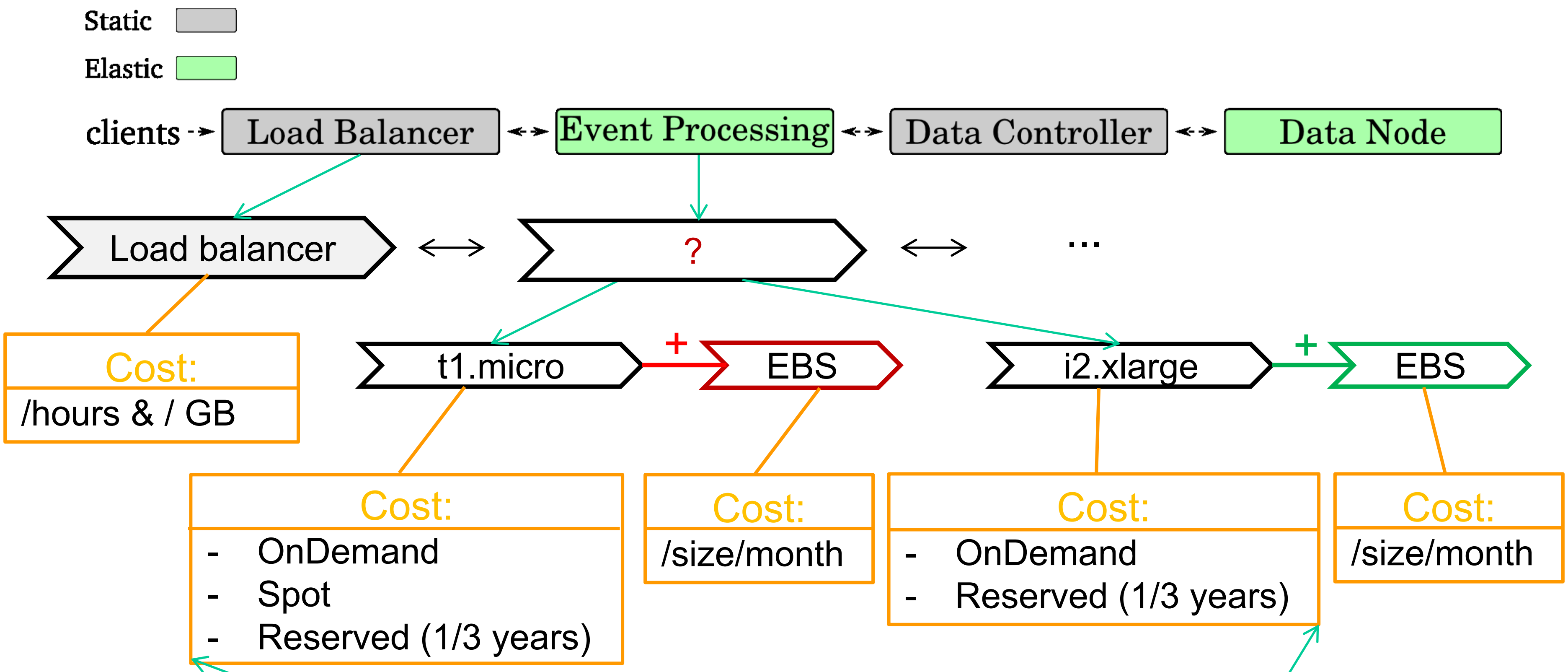
The functional capability of the unit and interface to access the function

Capabilities to be elastic under different requirements

Need to model/capture elasticity primitive operations


Elasticity Primitive Operations
Elasticity Capabilities
• • •
Virtual Machine
Storage
Management Platform
Service API
Add Memory
Add Cores
Remove Memory
• • •
Attach Smart Caching
Change Availability Region
Add Container
• • •
Change Backup Strategy
Change Licensing Scheme
Change Reservation Scheme
• • •
Add Virtual Machine
Customize Platform Service
Reconfigure Cloud Service
API.scaleUp(memory,size)
API.scaleDown(memory,size)
API.setBackup Period(time)
API.addNewContainer()
API. createVM(Flavor)
API.allocatePlatformComponent (artifacts,characteristics)
API.Reconfigure Service(Configx)
Elasticity Capabilities API

Programming: example of software-defined elastic systems (SESs)



Static ⬜

Elastic 🟩

Greater cost elasticity
Mandatory association with
EBS (Elastic Block Store)

Greater cost elasticity
Optional association with
EBS(Elastic Block Store)

Deploying, Control, Monitoring and Testing

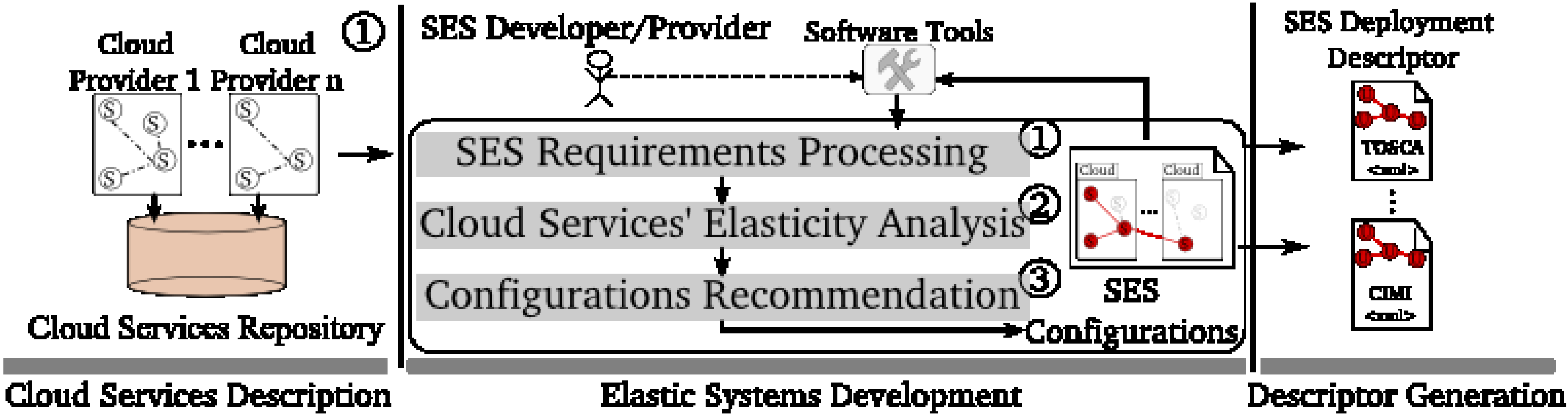
- Runtime configuration
  - Complex services at multiple software stacks (IaaS, PaaS and application)
    - Interfaces with different low-level deployment techniques
  - Different interactions between deploying and control and monitoring components
- Control elasticity
  - Using a high-level specification for specifying elasticity requirements, constraints and strategies
  - Based on SYBL/rSYBL ([CCGrid 2013])

Deploying, Control, Monitoring and Testing

- Elasticity monitoring and analysis
  - Utilize low-level metrics to build „Elasticity Space“ and analyze the elasticity based on such spaces (based on MELA – [CloudCom 2013])
  - Monitoring/analysis at multiple levels level (single unit, topology/group, and the whole service
- Testing elasticity
  - Using clouds to test cloud applications as well as to test elasticity properties of cloud applications [ASE2013, IC2014]

SOME TECHNIQUES – THE VIENNA WAY

Programming elasticity: Quantifying elasticity



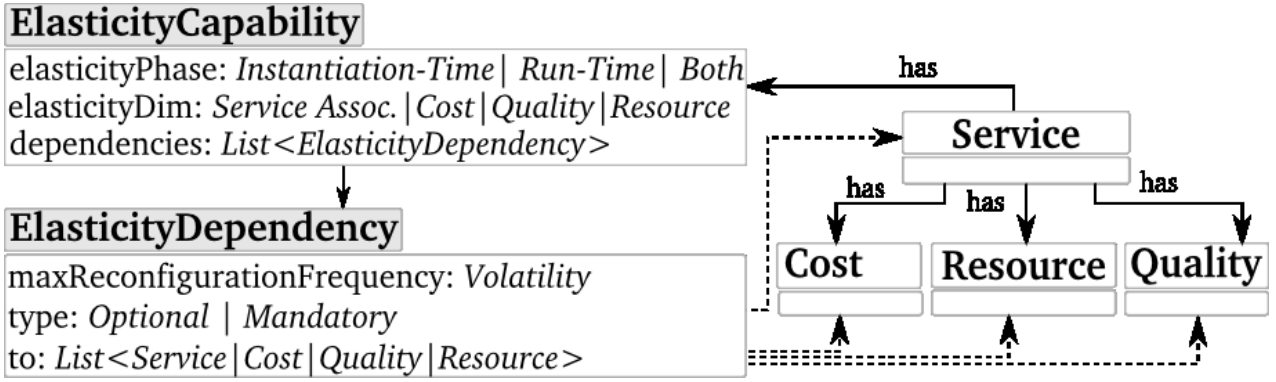
Cloud Provider 1
Cloud Provider n
①
Cloud Services Repository
Cloud Services Description
SES Developer/Provider
Software Tools
SES Requirements Processing ①
Cloud Services' Elasticity Analysis ②
Configurations Recommendation ③
Cloud
Cloud
SES Configurations
Elastic Systems Development
SES Deployment Descriptor
TOSCA <xml>
CIMI <xml>
Descriptor Generation

- Quantifying functions and configuration recommendation functions can be used by software development tools

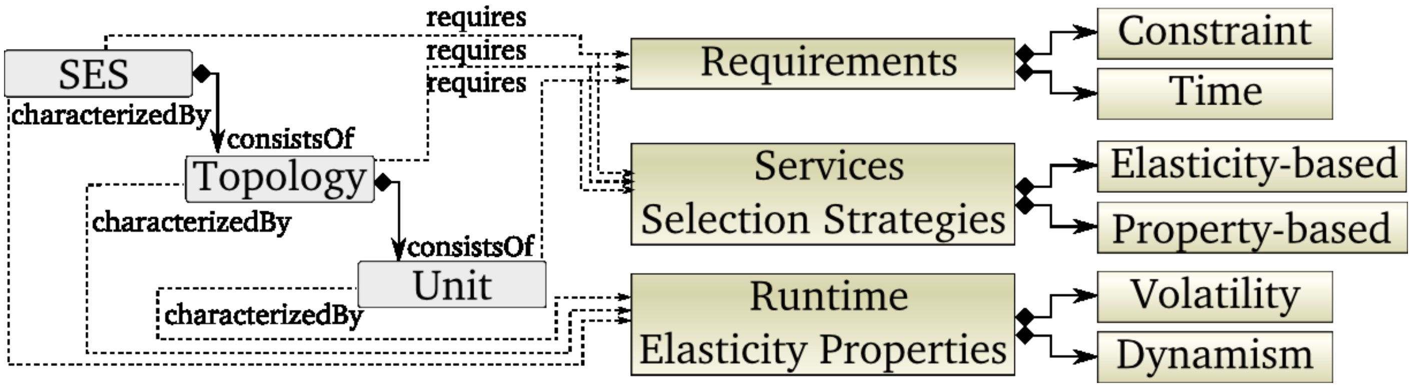
Daniel Moldovan, Georgiana Copil, Hong-Linh Truong, Schahram Dustdar, QUELLE – a Framework for Accelerating the Development of Elastic Systems, ESOCC 2014. September 2014

Programming Elasticity: Recommend elastic service units

Cloud ecosystems


ElasticityCapability
elasticityPhase: Instantiation-Time | Run-Time | Both
elasticityDim: Service Assoc. | Cost | Quality | Resource
dependencies: List<ElasticityDependency>
ElasticityDependency
maxReconfigurationFrequency: Volatility
type: Optional | Mandatory
to: List<Service | Cost | Quality | Resource>
has
Service
has
has
has
Cost
Resource
Quality

Elastic services to be developed



Daniel Moldovan, Georgiana Copil, Hong-Linh Truong, Schahram Dustdar, QUELLE – a Framework for Accelerating the Development of Elastic Systems, ESOCC 2014. September 2014

Programming Elasticity: Elasticity Quantifying functions

Elasticity Phase Quantification Coefficients

$$ElPhaseQ(p) = \begin{cases} v_i : if\ p = Instantiation\ Time \\ v_r : if\ p = Run\ Time \\ v_{ir} : if\ p = Both \end{cases}$$

Elasticity Dependency Type Quantification Coefficients

$$ElDependencyQ(dep) = \begin{cases} v_o : if\ dep = Optional\ Association \\ v_m : if\ dep = Mandatory\ Association \end{cases}$$

Elasticity Dependency Volatility Quantification Coefficients

$$VolatilityQ(dep)$$

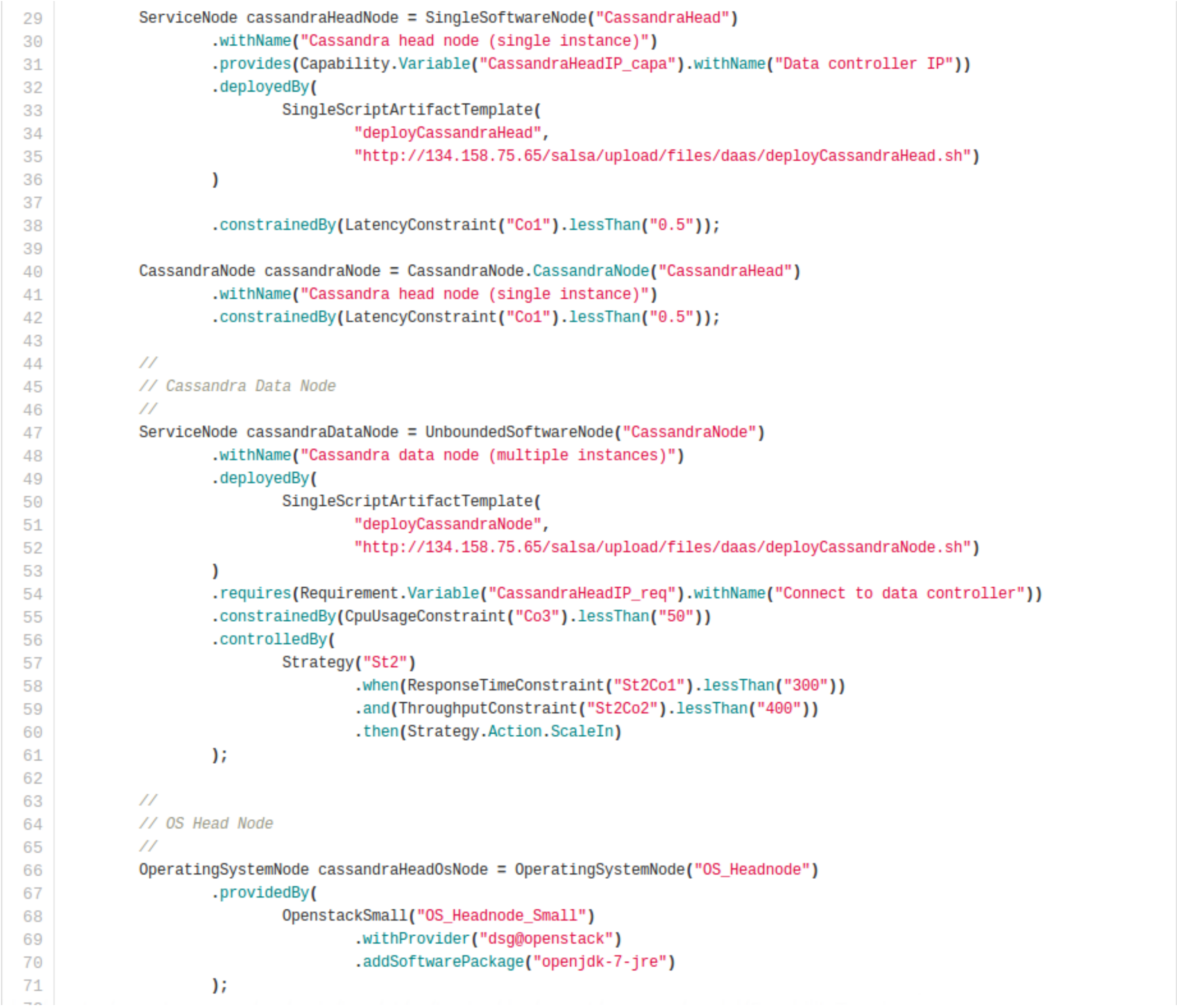
Elasticity Capability Quantification Function

$$ECQ(C) = ElPhaseQ(C.phase) * \sum_{dep \in C.dependencies} VolatilityQ(dep) * ElDependencyQ(dep)$$

Elasticity Quantification Function

$$EQ(S) = \sum_{D \in cost, quality, resource,} W(D) * \sum_{C \in D.capabilities} ECQ(C)$$

Programming: domain specific languages

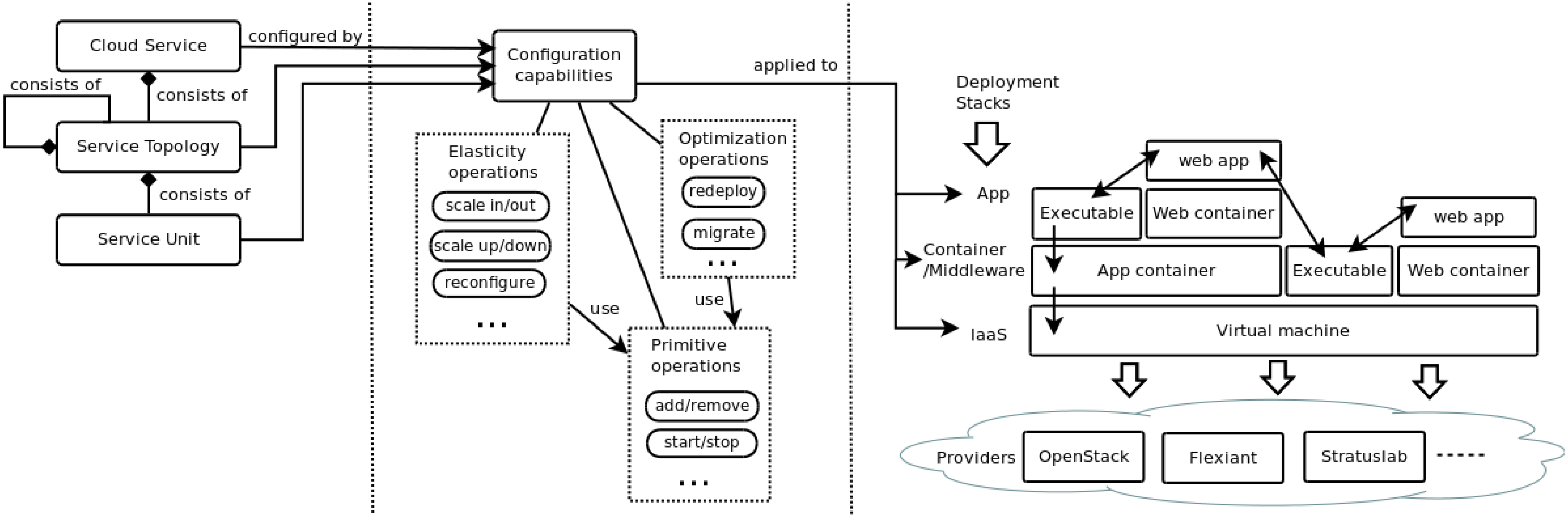


```
ServiceNode cassandraHeadNode = SingleSoftwareNode("CassandraHead")
        .withName("Cassandra head node (single instance)")
        .provides(Capability.Variable("CassandraHeadIP_capa").withName("Data controller IP"))
        .deployedBy(
                SingleScriptArtifactTemplate(
                        "deployCassandraHead",
                        "http://134.158.75.65/salsa/upload/files/daas/deployCassandraHead.sh")
        )

        .constrainedBy(LatencyConstraint("Co1").lessThan("0.5"));

CassandraNode cassandraNode = CassandraNode.CassandraNode("CassandraHead")
        .withName("Cassandra head node (single instance)")
        .constrainedBy(LatencyConstraint("Co1").lessThan("0.5"));

//
// Cassandra Data Node
//
ServiceNode cassandraDataNode = UnboundedSoftwareNode("CassandraNode")
        .withName("Cassandra data node (multiple instances)")
        .deployedBy(
                SingleScriptArtifactTemplate(
                        "deployCassandraNode",
                        "http://134.158.75.65/salsa/upload/files/daas/deployCassandraNode.sh")
        )
        .requires(Requirement.Variable("CassandraHeadIP_req").withName("Connect to data controller"))
        .constrainedBy(CpuUsageConstraint("Co3").lessThan("50"))
        .controlledBy(
                Strategy("St2")
                        .when(ResponseTimeConstraint("St2Co1").lessThan("300"))
                        .and(ThroughputConstraint("St2Co2").lessThan("400"))
                        .then(Strategy.Action.ScaleIn)
        );

//
// OS Head Node
//
OperatingSystemNode cassandraHeadOsNode = OperatingSystemNode("OS_Headnode")
        .providedBy(
                OpenstackSmall("OS_Headnode_Small")
                        .withProvider("dsg@openstack")
                        .addSoftwarePackage("openjdk-7-jre")
        );
```

The programmer does not want to deal with all types of nodes/artifacts

Runtime deployment and configuration

- Multi-cloud, multi-stack, complex topologies
  - Well-defined APIs for manipulating and provisioning objects
  - Support different types of objects, e.g., VMs, services, IoT sensors, gateways,



High level Elasticity control

- SYBL (Simple Yet Beautiful Language) for specifying elasticity requirements
- SYBL-supported requirement levels
  - Cloud Service Level
  - Service Topology Level
  - Service Unit Level
  - Relationship Level
  - Programming/Code Level

```
#SYBL.CloudServiceLevel
Cons1: CONSTRAINT responseTime < 5 ms
Cons2: CONSTRAINT responseTime < 10 ms
WHEN nbOfUsers > 10000
Str1: STRATEGY CASE fulfilled(Cons1) OR
fulfilled(Cons2): minimize(cost)

#SYBL.ServiceUnitLevel
Str2: STRATEGY CASE ioCost < 3 Euro :
maximize( dataFreshness )

#SYBL.CodeRegionLevel
Cons4: CONSTRAINT dataAccuracy>90%
AND cost<4 Euro
```

Georgiana Copil, Daniel Moldovan, Hong-Linh Truong, Schahram Dustdar, **"SYBL: an Extensible Language for Controlling Elasticity in Cloud Applications"**, 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), May 14-16, 2013, Delft, Netherlands

Configurations for multiclouds

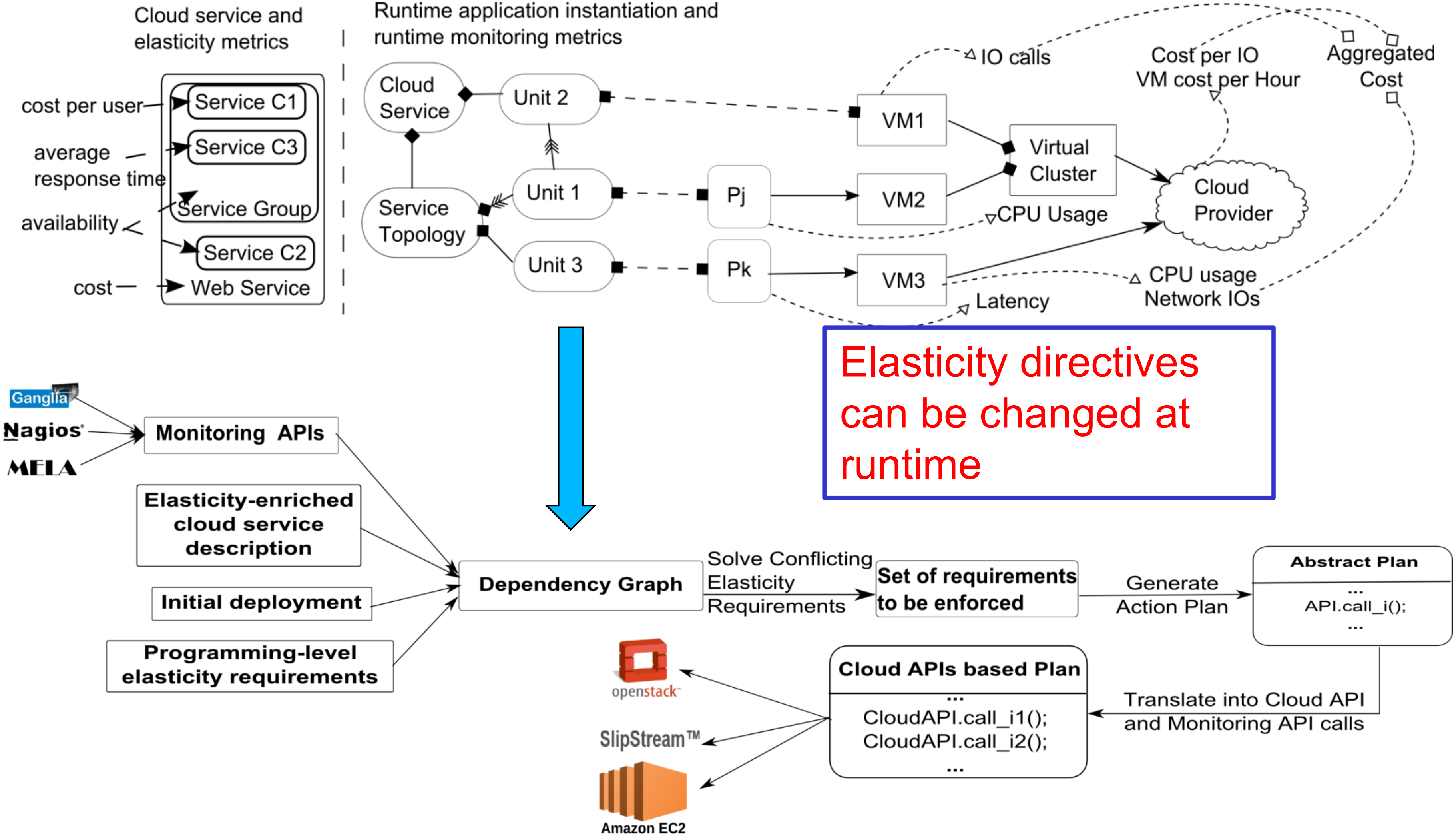
Configuraring low-level
Plug-ins to work with multiple clouds

```
MultipleEnforcementPlugins =
flexiant:at.ac.tuwien.dsg.rSybl.cloudInteractionUnit.enfor
cementPlugins.flexiant.EnforcementFlexiantAPI,
openstack:at.ac.tuwien.dsg.rSybl.cloudInteractionUnit.en
forcementPlugins.openstack.EnforcementOpenstackAPI
```

Configuraring and capturing elasticity primitive operations associated with service units

```
 ..
   <ServiceElasticityPrimitives id="FCO"
ServiceProvider="Flexiant FCO">
        <ElasticityPrimitive id="ScaleIn" name="Remove
VM" parameters=""/>
        <ElasticityPrimitive id="ScaleOut" name="Create
new VM" parameters=""/>
        <ElasticityPrimitive id="AllocateIP" name="Allocate
public IP" parameters="UUID"/>
        <ElasticityPrimitive id="AttachDisk" name="Attach
NewDisk" parameters="UUID">
…
            <PrimitiveDependency
dependencyType="AFTER_ENFORCEMENT"
primitiveID="Reboot"/>
         …
```

Mapping Services Structures to Elasticity Metrics

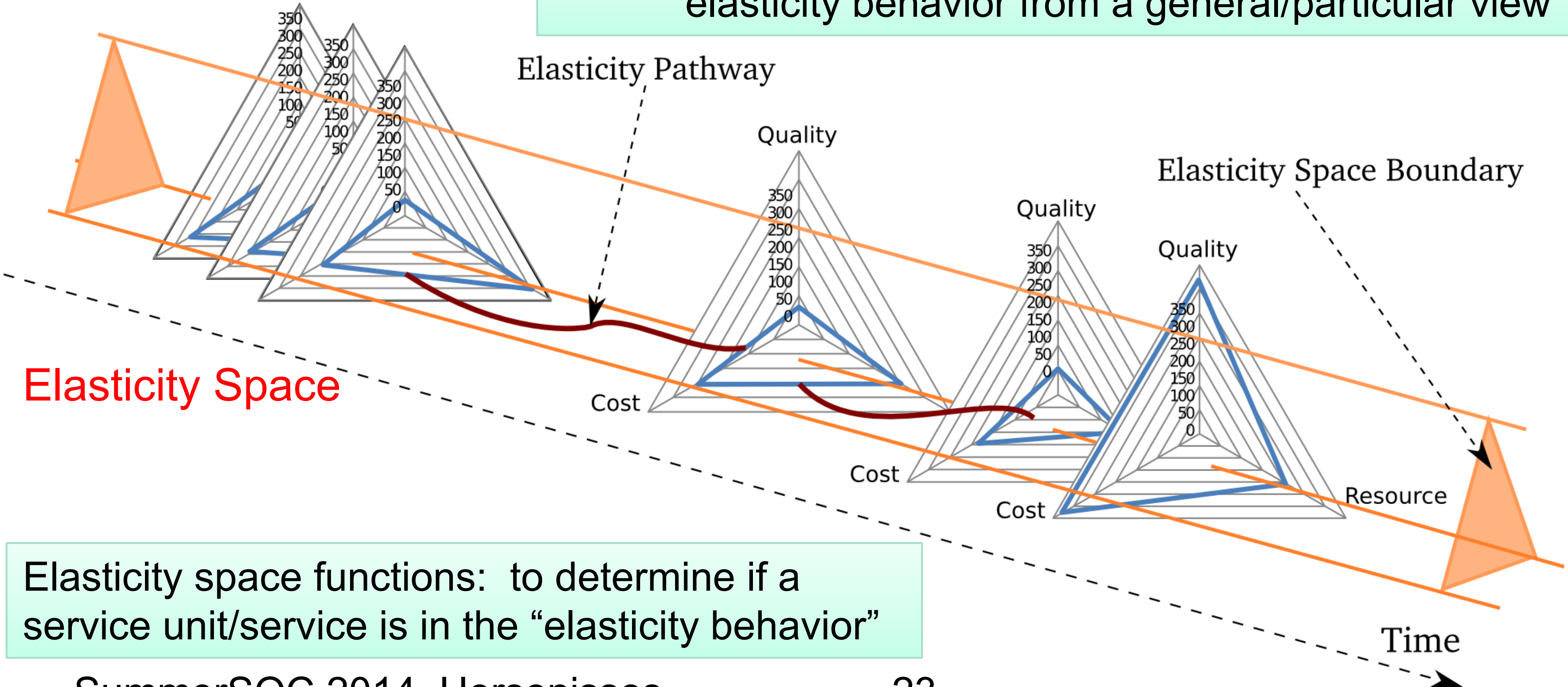


Elasticity directives can be changed at runtime

Elasticity Analysis for Cloud Services

Moldovan D., G. Copil,Truong H.-L., Dustdar S. (2013). **MELA: Monitoring and Analyzing Elasticity of Cloud Service. CloudCom 2013**

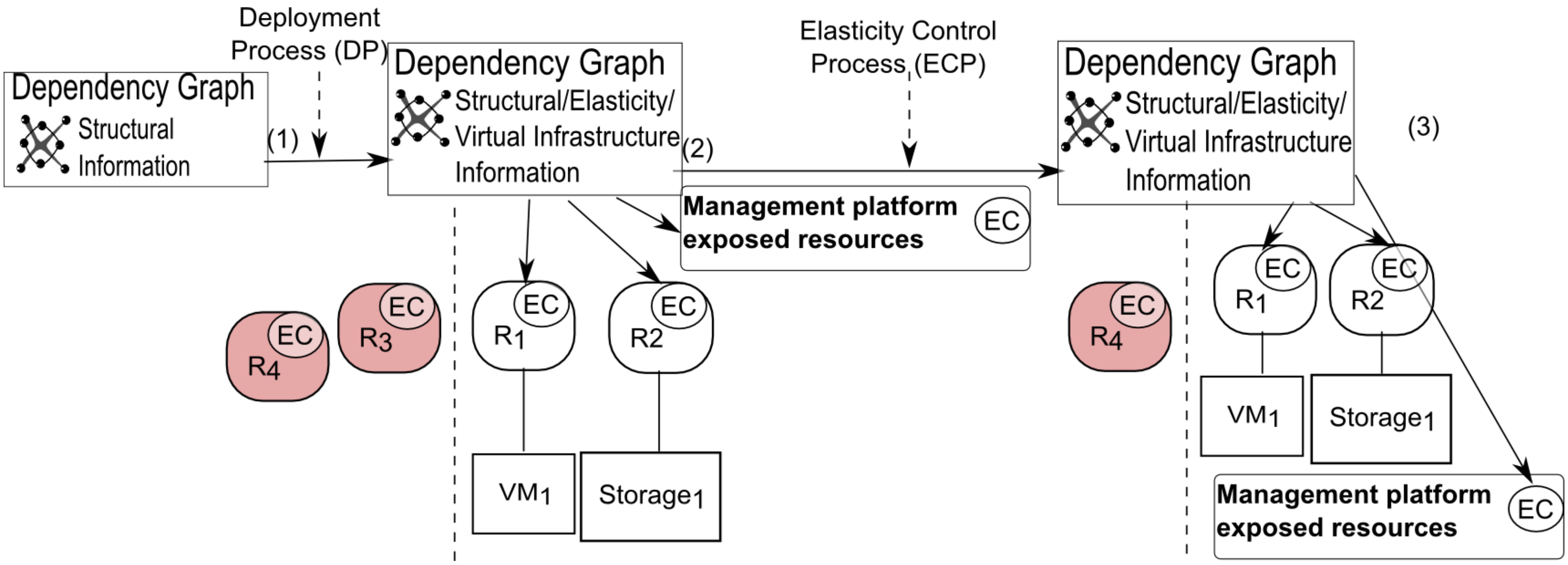
Elasticity Pathway functions: to characterize the elasticity behavior from a general/particular view



Elasticity space functions: to determine if a service unit/service is in the "elasticity behavior"

Learning elasticity behavior

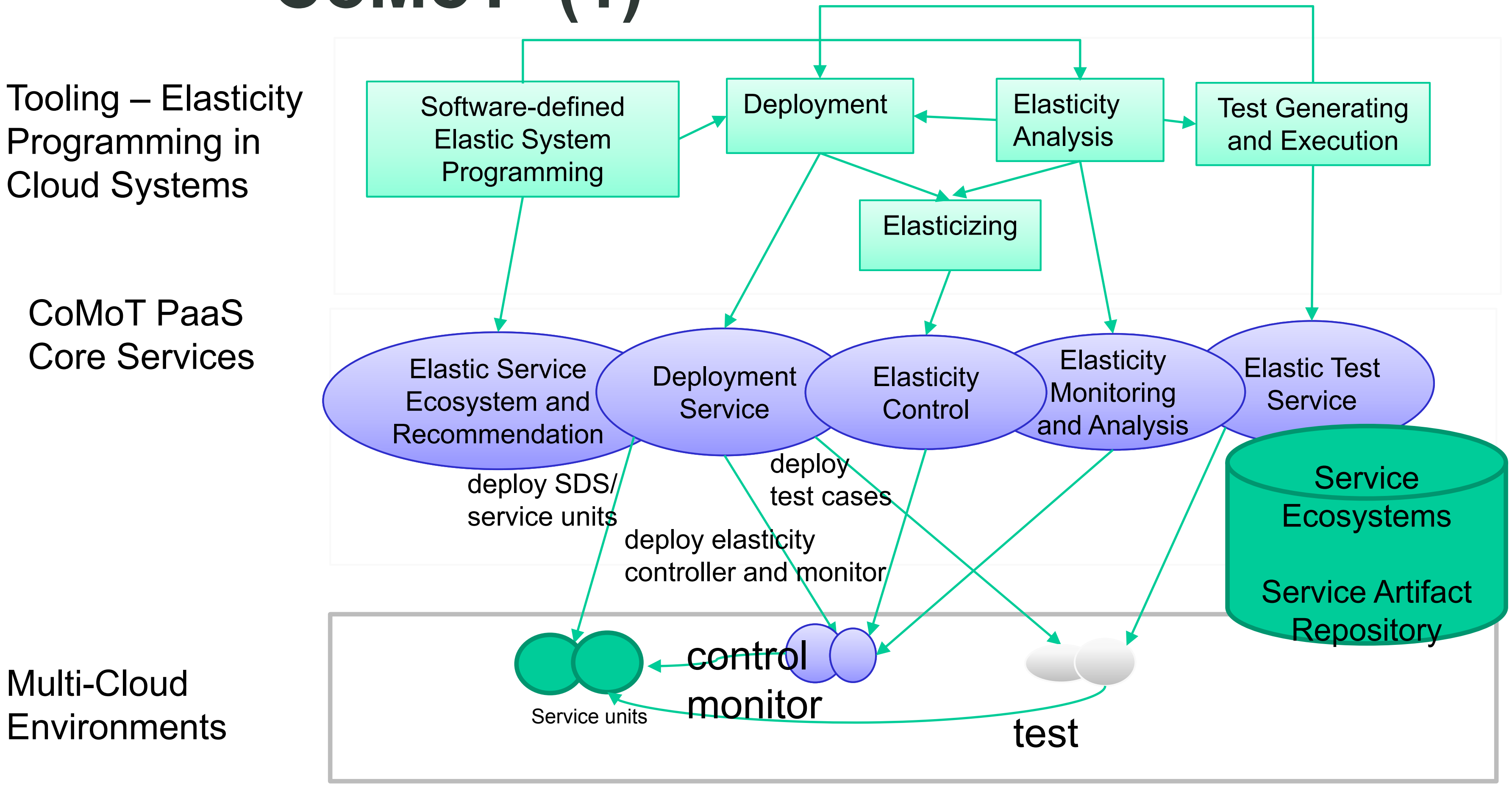
We need to learn elasticity behaviors and use them for controlling cloud services



Deployment Process (DP)
Dependency Graph
Structural Information
(1)
Dependency Graph
Structural/Elasticity/ Virtual Infrastructure Information
Elasticity Control Process (ECP)
(2)
Dependency Graph
Structural/Elasticity/ Virtual Infrastructure Information
(3)
Management platform exposed resources
EC
R4
EC
R3
EC
R1
EC
R2
EC
VM1
Storage1
R4
EC
R1
EC
R2
EC
VM1
Storage1
Management platform exposed resources
EC

Georgiana Copil, Demetris Trihinas, Hong-Linh Truong, Daniel Moldovan, George Pallis, Schahram Dustdar, Marios Dikaiakos, ADVISE – a Framework for Evaluating Cloud Service Elasticity Behavior. May 2014. On Submission

CoMoT (1)


Tooling – Elasticity Programming in Cloud Systems
Software-defined Elastic System Programming
Deployment
Elasticity Analysis
Test Generating and Execution
Elasticizing
CoMoT PaaS Core Services
Elastic Service Ecosystem and Recommendation
Deployment Service
Elasticity Control
Elasticity Monitoring and Analysis
Elastic Test Service
Service Ecosystems
Service Artifact Repository
deploy SDS/ service units
deploy test cases
deploy elasticity controller and monitor
Multi-Cloud Environments
Service units
control
monitor
test

Hong-Linh Truong et al., **"CoMoT – A Platform-as-a-Service for Elasticity in the Cloud"**, IEEE International Workshop on the Future of PaaS. Colocated with IC2E 2014

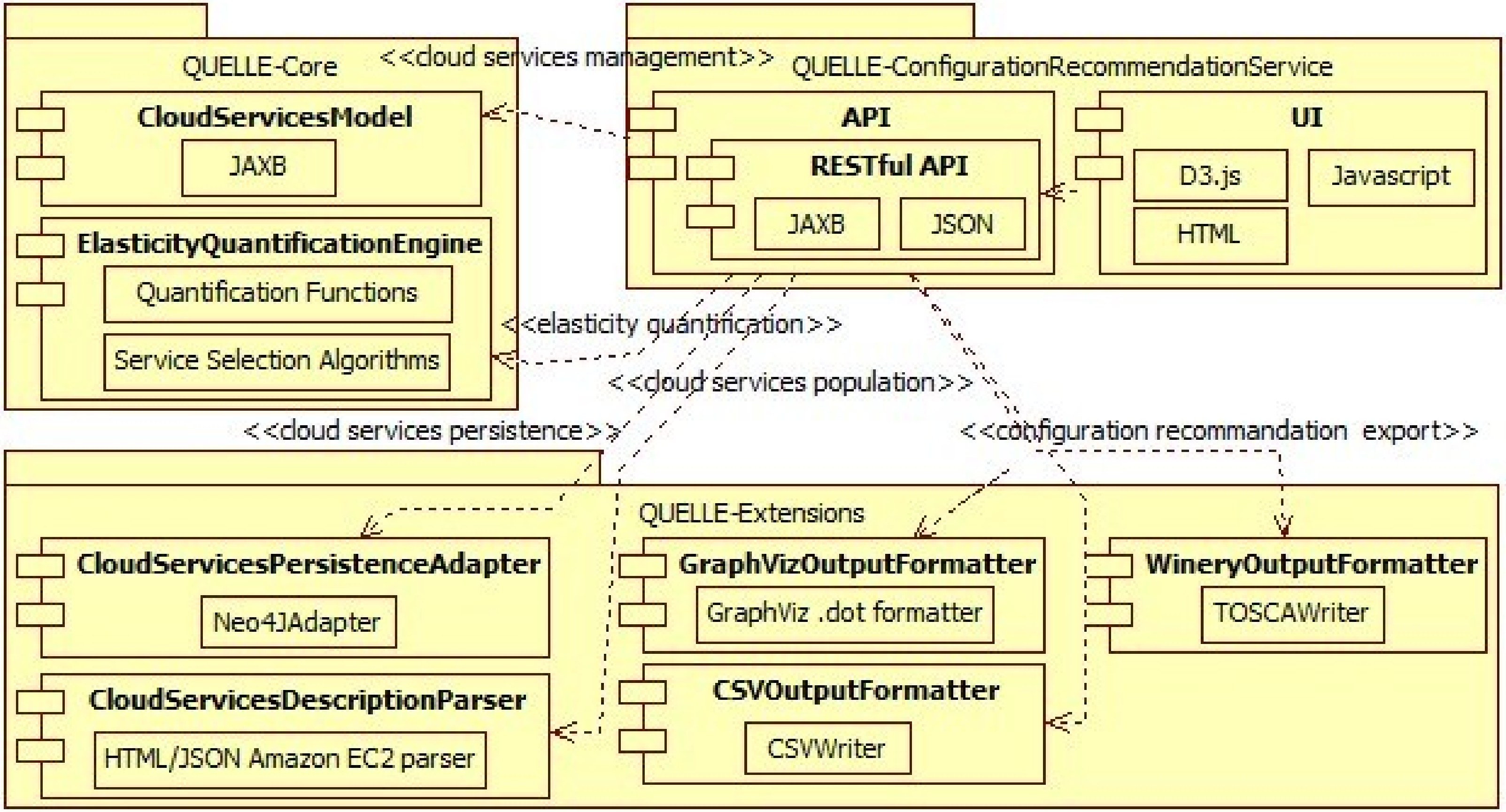
CoMoT (2)

- CoMoT is built atop:
  - QUELLE, rSYBL, MELA, SALSA, AutoCles

- Work on multi-cloud environments
  - Parts of complex applications are deployed in different clouds

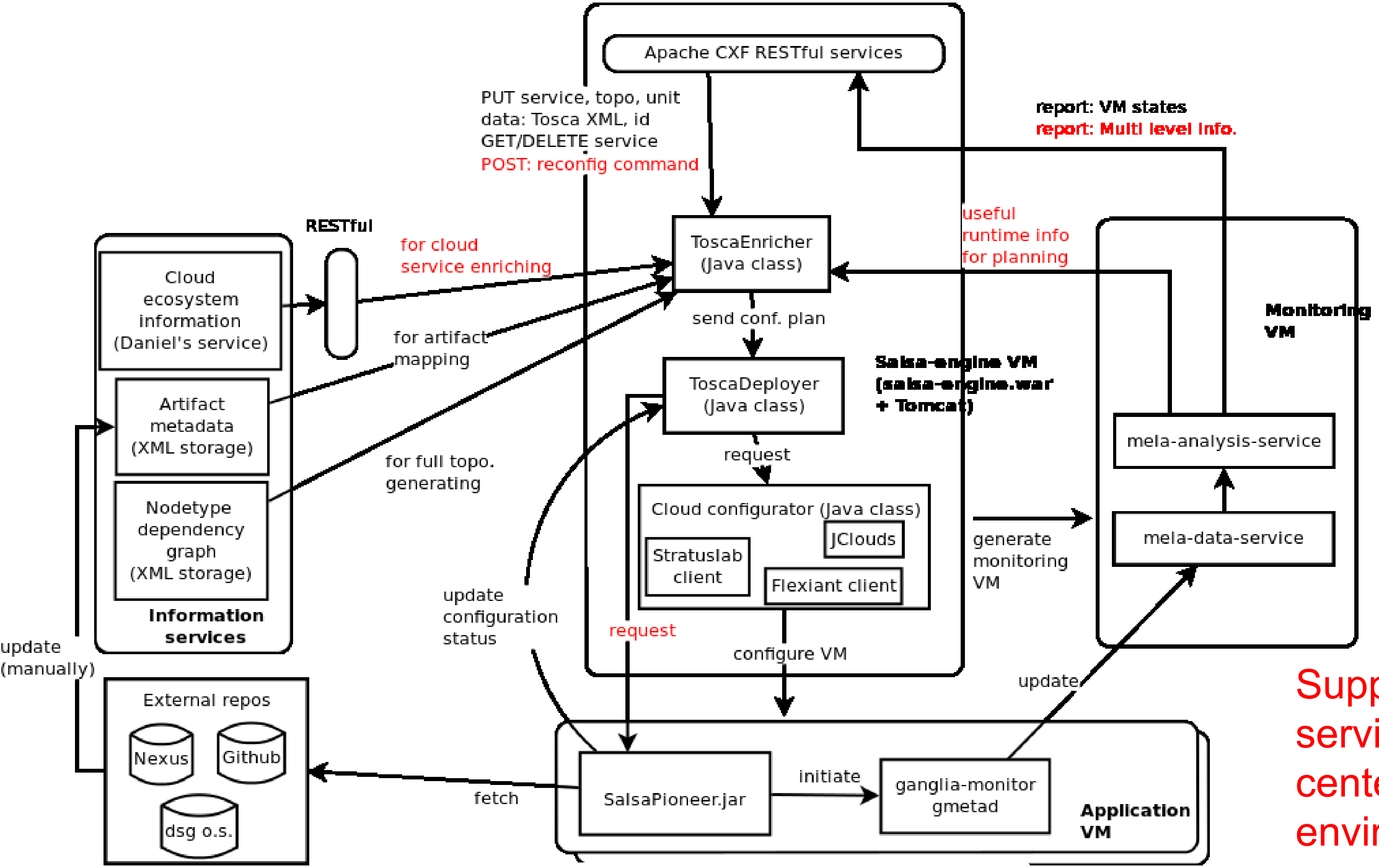
- GIT: https://github.com/tuwiendsg and https://github.com/whummer/AUToCLES

QUELLE – QUantifying ELasticity utiLity Engine


QUELLE-Core
<<cloud services management>>
QUELLE-ConfigurationRecommendationService
CloudServicesModel
JAXB
ElasticityQuantificationEngine
Quantification Functions
Service Selection Algorithms
API
RESTful API
JAXB
JSON
UI
D3.js
Javascript
HTML
<<elasticity quantification>>
<<cloud services population>>
<<cloud services persistence>>
<<configuration recommandation export>>
QUELLE-Extensions
CloudServicesPersistenceAdapter
Neo4JAdapter
CloudServicesDescriptionParser
HTML/JSON Amazon EC2 parser
GraphVizOutputFormatter
GraphViz .dot formatter
CSVOutputFormatter
CSVWriter
WineryOutputFormatter
TOSCAWriter

Daniel Moldovan, Georgiana Copil, Hong-Linh Truong, Schahram Dustdar, QUELLE – a Framework for Accelerating the Development of Elastic Systems, ESOCC 2014. September 2014

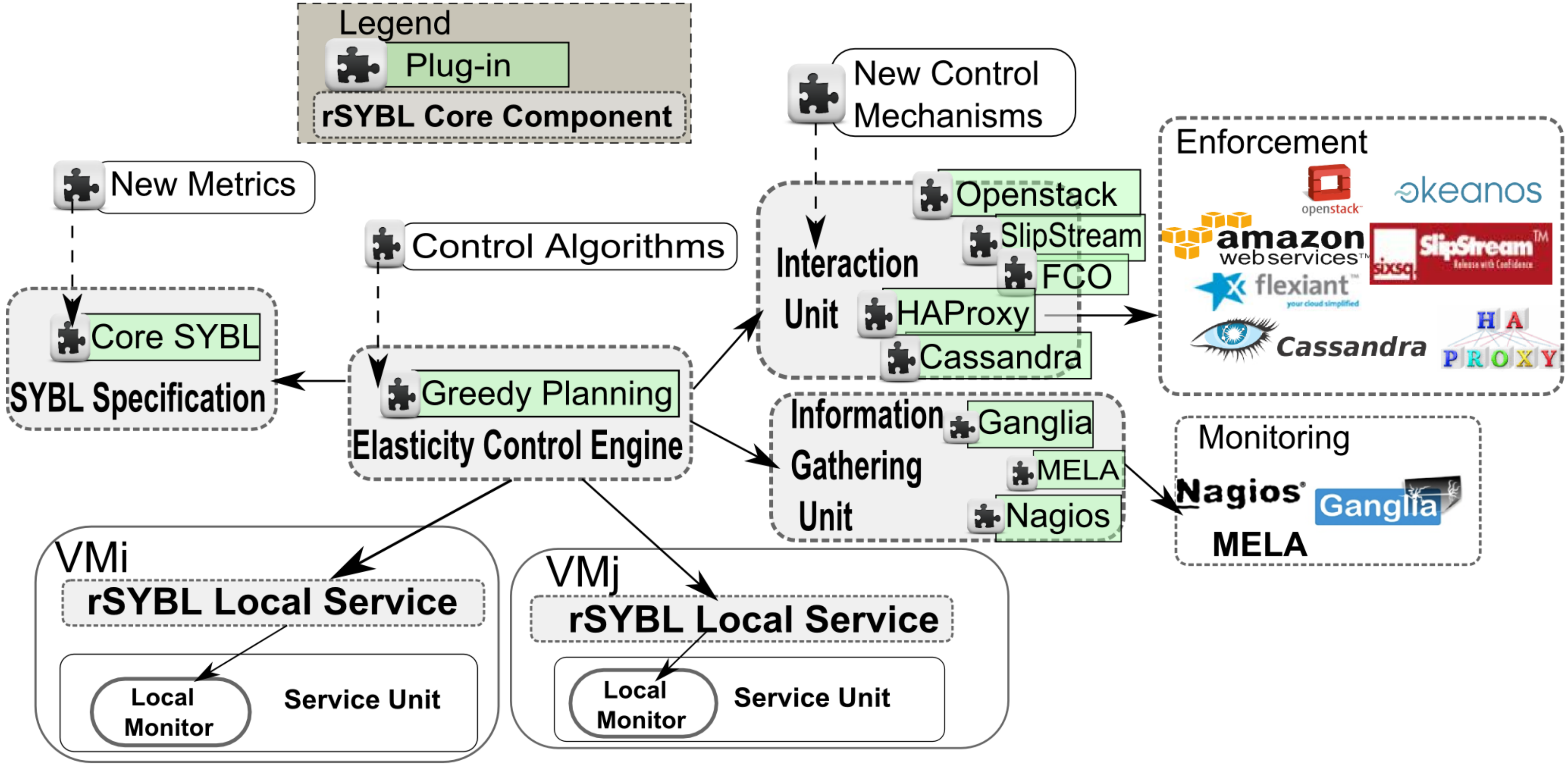
SALSA - the deployment framework



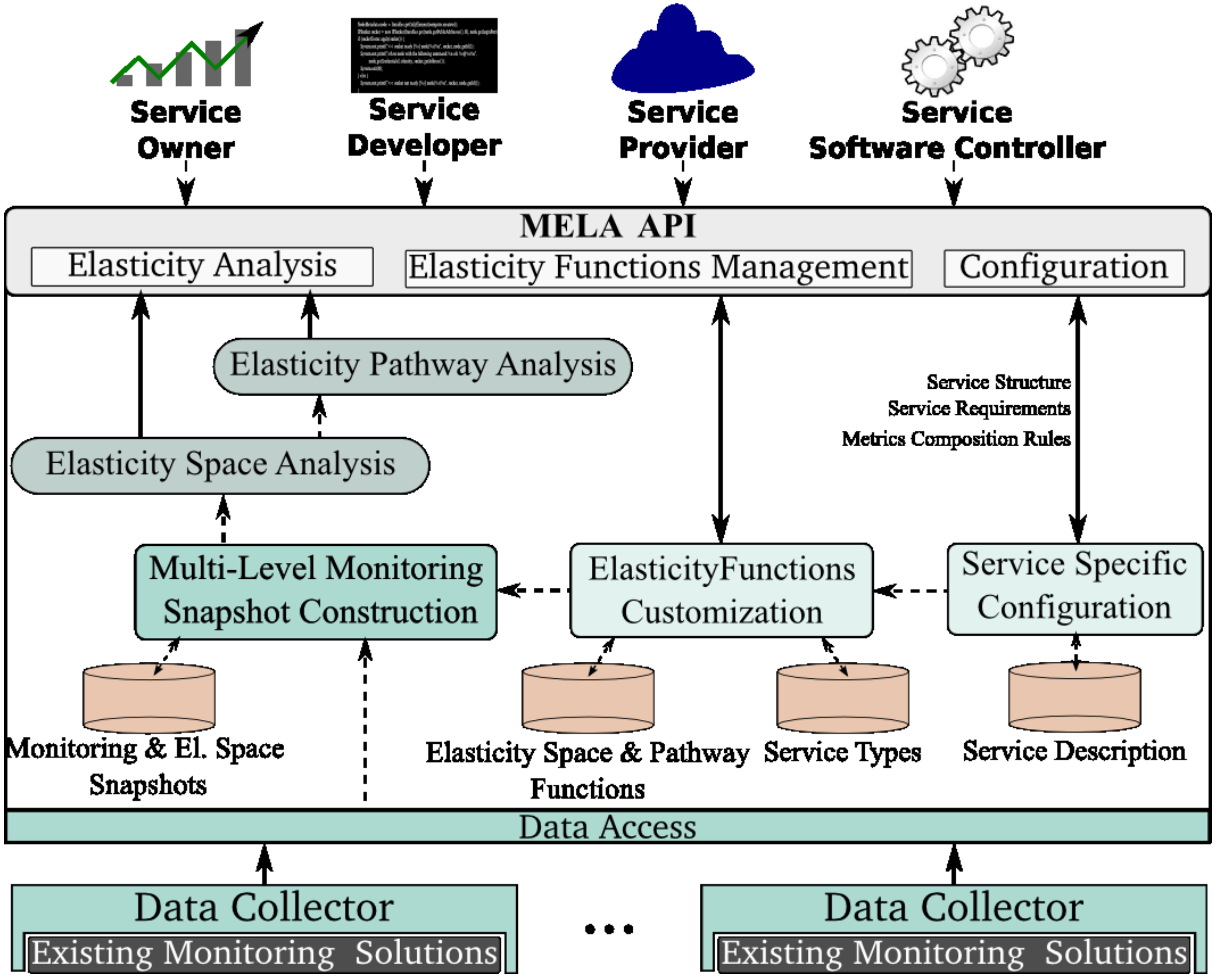
Support cloud services at data centers and IoT environments

https://github.com/tuwiendsg/SALSA

rSYBL – Elasticity Control Engine



MELA -- Elasticity Monitoring as a Service



Service Owner

Service Developer

Service Provider

Service Software Controller

MELA API

Elasticity Analysis

Elasticity Functions Management

Configuration

Elasticity Pathway Analysis

Service Structure

Service Requirements

Metrics Composition Rules

Elasticity Space Analysis

Multi-Level Monitoring Snapshot Construction

ElasticityFunctions Customization

Service Specific Configuration

Monitoring & El. Space Snapshots

Elasticity Space & Pathway Functions

Service Types

Service Description

Data Access

Data Collector

Existing Monitoring Solutions

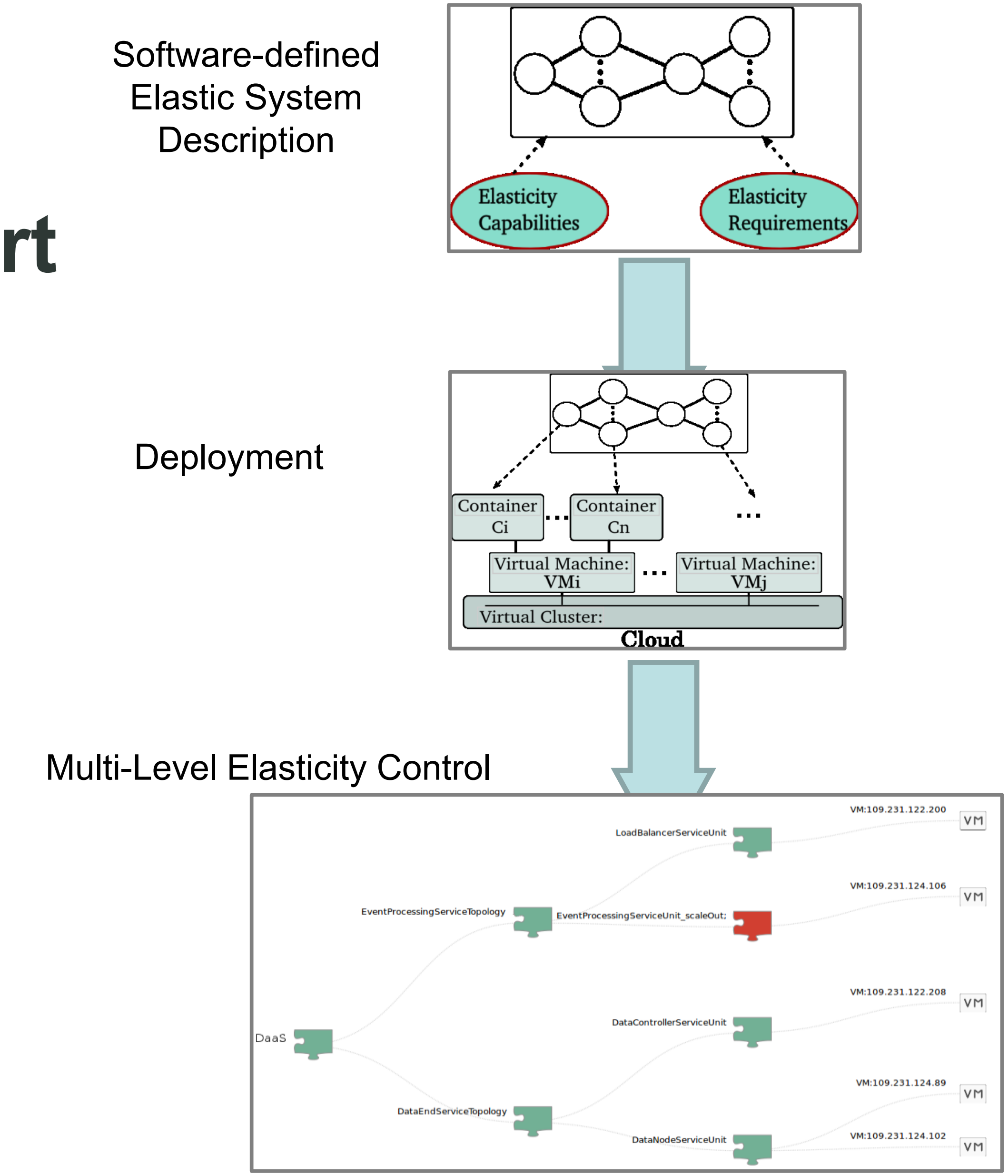
...

Data Collector

Existing Monitoring Solutions

Daniel Moldovan, Georgiana Copil, Hong-Linh Truong, Schahram Dustdar, MELA: Monitoring and Analyzing Elasticity of Cloud Services. CloudCom 2013

CoMoT – Support all phases for elasticity engineering of cloud software services



Software-defined Elastic System Description

Deployment

Multi-Level Elasticity Control

SEE THE FOLLOW-UP DEMO

Conclusions and future work

- Native cloud applications need novel toolsets
  - Design, deployment, control, monitoring and testing of elasticity in interwoven engineering phases
  - CoMoT introduces concepts of elastic objects and fundamental building blocks for engineering an end-to-end elasticity for cloud services
- Future works
  - DSL for elastic objects
  - Further work on hot deployment and configuration under elasticity control
  - Testing elasticity dependencies

Thanks for your attention!

Hong-Linh Truong

Distributed Systems Group
TU Wien

truong@dsg.tuwien.ac.at
dsg.tuwien.ac.at/research/viecom