

c-Eclipse: An Open-Source Management Framework for Cloud Applications

Chrystalla Sofokleous, Nicholas Loulloudes, Demetris Trihinas,
George Pallis, Marios D. Dikaiakos

{stalosof, loulloudes.n, trihinas, gpallis, mdd}@cs.ucy.ac.cy

Euro-Par 2014, Porto, Portugal, 28 August 2014

Cloud Application Management Challenges

- Increasing complexity of Cloud applications
 - Specially when considering their dynamic nature
 - Description of applications is a complex endeavor
- Growing number of IaaS-providers
 - Need to find the best possible vendor to host an application
 - Might be required to describe an application for deployment over different infrastructures

Cloud Application Management Challenges

- Frameworks have been developed to ease the description & deployment of applications over Cloud infrastructures
 - Most frameworks are vendor-specific
 - Locking their users to the specific vendors
- Describing an application for deployment over alternative Clouds is challenging

Existing Cloud Application Management Frameworks

PROPRIETARY FRAMEWORKS

	Portability	Cloud Platform Independent	Unified Environment	Open Specifications	Open Source	Elasticity Specification
Amazon Cloud Formation			✓			✓
VMWare's vFabric			✓			
Oracle's OVAB	✓			✓		
Service Mesh's Agility Platform	✓	✓	✓			✓
Juju	✓	✓	✓		✓	
Winery	✓	✓		✓	✓	
Cloudify	✓	✓		✓	✓	

Proprietary Cloud Application Management Frameworks

	Portability	Cloud Platform Independent	Unified Environment	Open Specifications	Open Source	Elasticity Specification
Amazon Cloud Formation			✓			✓
VMWare's vFabric			✓			
Oracle's OVAB	✓			✓		

Benefits

- Well integrated with underlying platform
 - Easier application orchestration
- Ease of use
 - Good documentation & User support



ORACLE
VIRTUAL ASSEMBLY BUILDER

amazon web services CloudFormation

vmware vFabric

Limitations

- Commercial
- Lock users to specific vendors/technologies

Existing Cloud Application Management Frameworks

	Portability	Cloud Platform Independent	Unified Environment	Open Specifications	Open Source	Elasticity Specification
Amazon Cloud Formation			✓			✓
VMWare's vFabric			✓			
Oracle's OVAB	✓			✓		
Service Mesh's Agility Platform	✓	✓	✓			✓
Juju	✓	✓	✓		✓	
Winery	✓	✓		✓	✓	
Cloudify	✓	✓		✓	✓	

GENERIC FRAMEWORKS

Generic Cloud Application Management Frameworks

	Portability	Cloud Platform Independent	Unified Environment	Open Specifications	Open Source	Elasticity Specification
Service Mesh's Agility Platform	✓	✓	✓			✓
Juju	✓	✓	✓		✓	
Winery	✓	✓		✓	✓	
Cloudify	✓	✓		✓	✓	

Benefits

- Applications are portable across different Cloud infrastructures

winery



Limitations

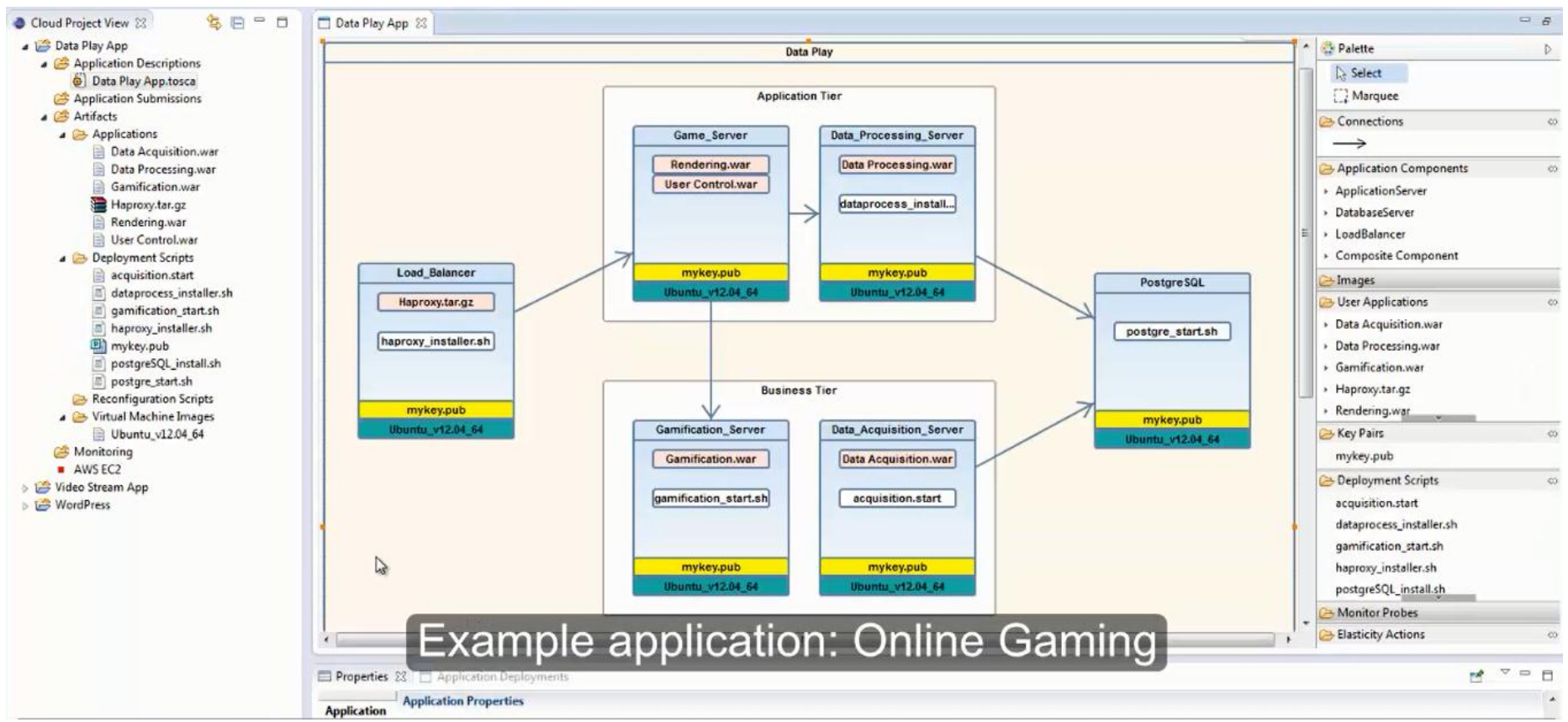
- Support limited number of Cloud platforms
 - Financial overhead when migrating across providers
- Limited or no elasticity support i.e. only add/remove VM



c-Eclipse

The screenshot displays the c-Eclipse interface for a cloud project named 'VideoStream'. The central workspace shows a diagram titled 'VideoGenericDescription' with three application components: 'Load_Balancer', 'Application_Server', and 'NoSQL_Database'. The 'Load_Balancer' component contains 'HaProxy.tar' and 'haproxy_config.sh'. The 'Application_Server' component contains 'VideoService.war' and 'tomcat_config.sh'. An arrow points from the 'Application_Server' to the 'NoSQL_Database' component, which contains 'cassandra_config.sh'. The left sidebar shows a tree view of the project structure, including 'Application Descriptions', 'Artifacts', and 'Deployment Scripts'. The right sidebar shows a 'Palette' with various components and scripts. The bottom panel shows the 'Application Component Properties' for the 'Application_Server' component, including fields for 'Name', 'VM Image', 'VM Description', and 'Number of Instances' (Initial: 1, Min: 1, Max: 3).

c-Eclipse



c-Eclipse

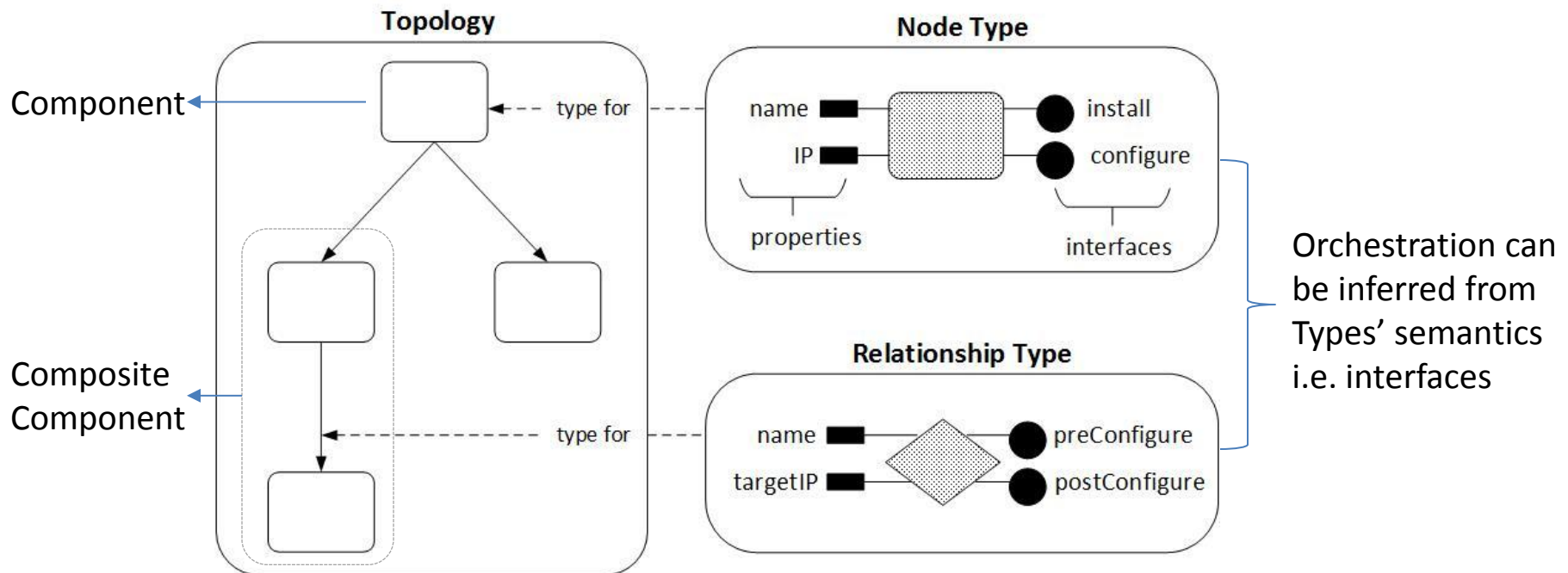


- ✓ Open-source (on top of Eclipse platform)
- ✓ Platform independent
 - ✓ Runs on any OS supported by Eclipse
- ✓ Intuitive graphical drag-and-drop UI
 - ✓ Low entry barrier for new end-users
- ✓ Adopts open Cloud specifications
 - ✓ Application portability
- ✓ Adopts open language for describing Cloud applications' elasticity requirements

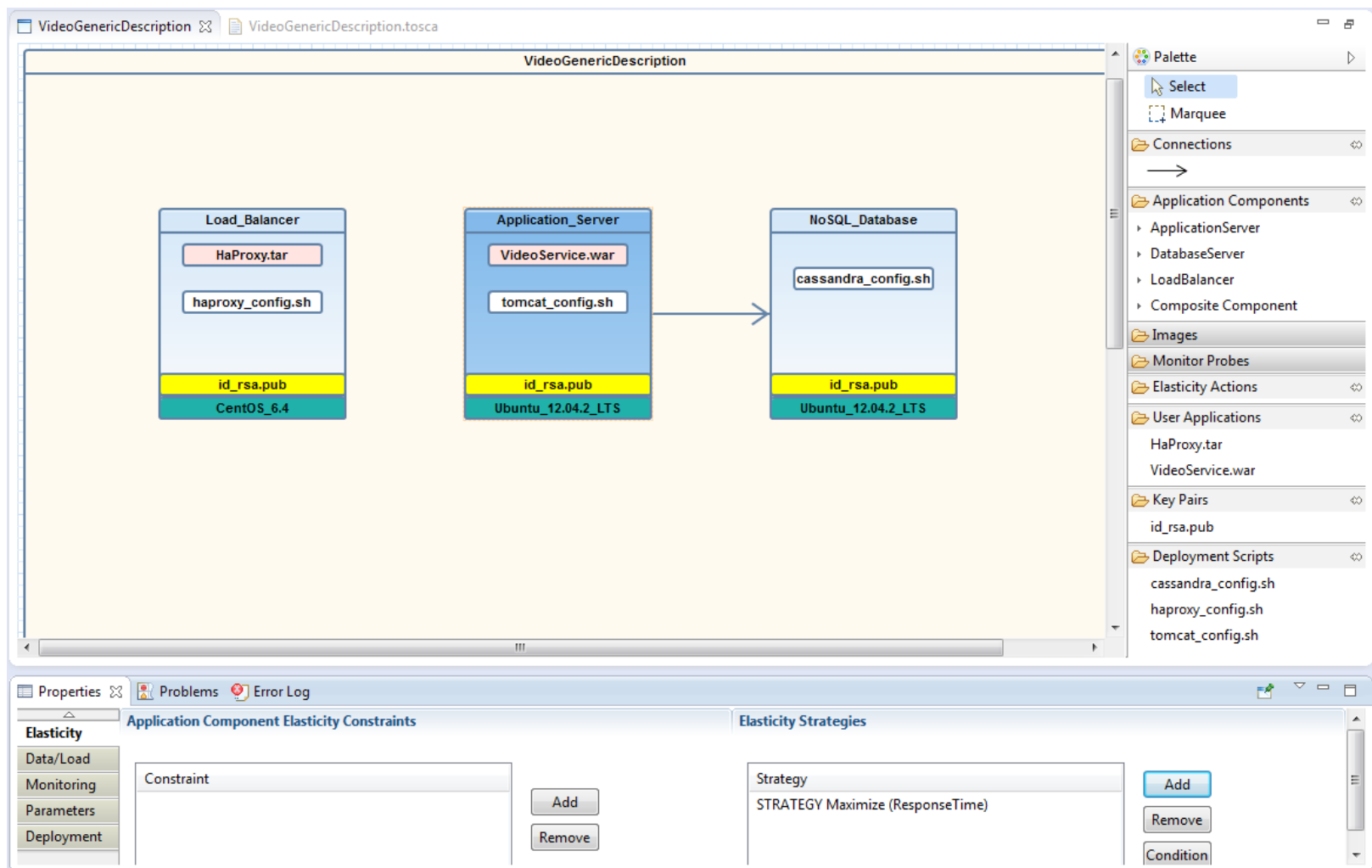
Background

OASIS TOSCA

- TOSCA provides a language to describe
 - Application components & their relationships (**topology**)
 - Application management procedures (**orchestration**)



Graphical TOSCA Modeling



TOSCA XML translation

```

VideoGenericDescription *VideoGenericDescription.tosca
<?xml version="1.0" encoding="UTF-8"?>
<tosca:Definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:elasticity="http://www.example.org/NewXMLSchema" xmlns:sybl="http://www.examp
  <tosca:ServiceTemplate id="hello" name="VideoGenericDescription">
    <tosca:BoundaryDefinitions xsi:type="elasticity:TBoundaryDefinitionsExtension"/>
    <tosca:TopologyTemplate>
      <tosca:NodeTemplate xsi:type="elasticity:TNodeTemplateExtension" id="C968712039" type="appserver" maxInstances="3" minInstances="1" name="Application_
        <tosca:Policies>
          <tosca:Policy name="STRATEGY Maximize (ResponseTime)" policyRef="C1198524356" policyType="sybl:Strategy"/>
        </tosca:Policies>
        <tosca:DeploymentArtifacts>
          <tosca:DeploymentArtifact artifactType="UA" name="VideoService.war"/>
          <tosca:DeploymentArtifact artifactType="SD" name="tomcat_config.sh"/>
          <tosca:DeploymentArtifact artifactType="KeyPair" name="id_rsa.pub"/>
          <tosca:DeploymentArtifact artifactType="VMI" name="Ubuntu_12.04.2_LTS"/>
        </tosca:DeploymentArtifacts>
      </tosca:NodeTemplate>
      <tosca:RelationshipTemplate id="R1946514275" type="Directed" name="Directed Relation">
        <tosca:SourceElement ref="C968712039"/>
        <tosca:TargetElement ref="C1198363563"/>
      </tosca:RelationshipTemplate>
    </tosca:TopologyTemplate>
  </tosca:ServiceTemplate>
  <tosca:PolicyTemplate id="C1198524356" type="sybl:Strategy">
    <tosca:Properties>
      <sybl:SYBLElasticityRequirementsDescription>
        <SYBLSpecification>
          <Strategy Id="hi">
            <ToEnforce ActionName="Maximize (ResponseTime)" Parameter="hi2"/>
          </Strategy>
        </SYBLSpecification>
      </sybl:SYBLElasticityRequirementsDescription>
    </tosca:Properties>
  </tosca:PolicyTemplate>
  <tosca:NodeTemplate xsi:type="elasticity:TNodeTemplateExtension" id="C1198363563" type="dbserver" maxInstances="-1" minInstances="-1" name="NoSQL_Database

```

OASIS TOSCA

- TOSCA specifies an **exchange format** to package Cloud applications, named **CSAR**
 - Cloud Service Archive

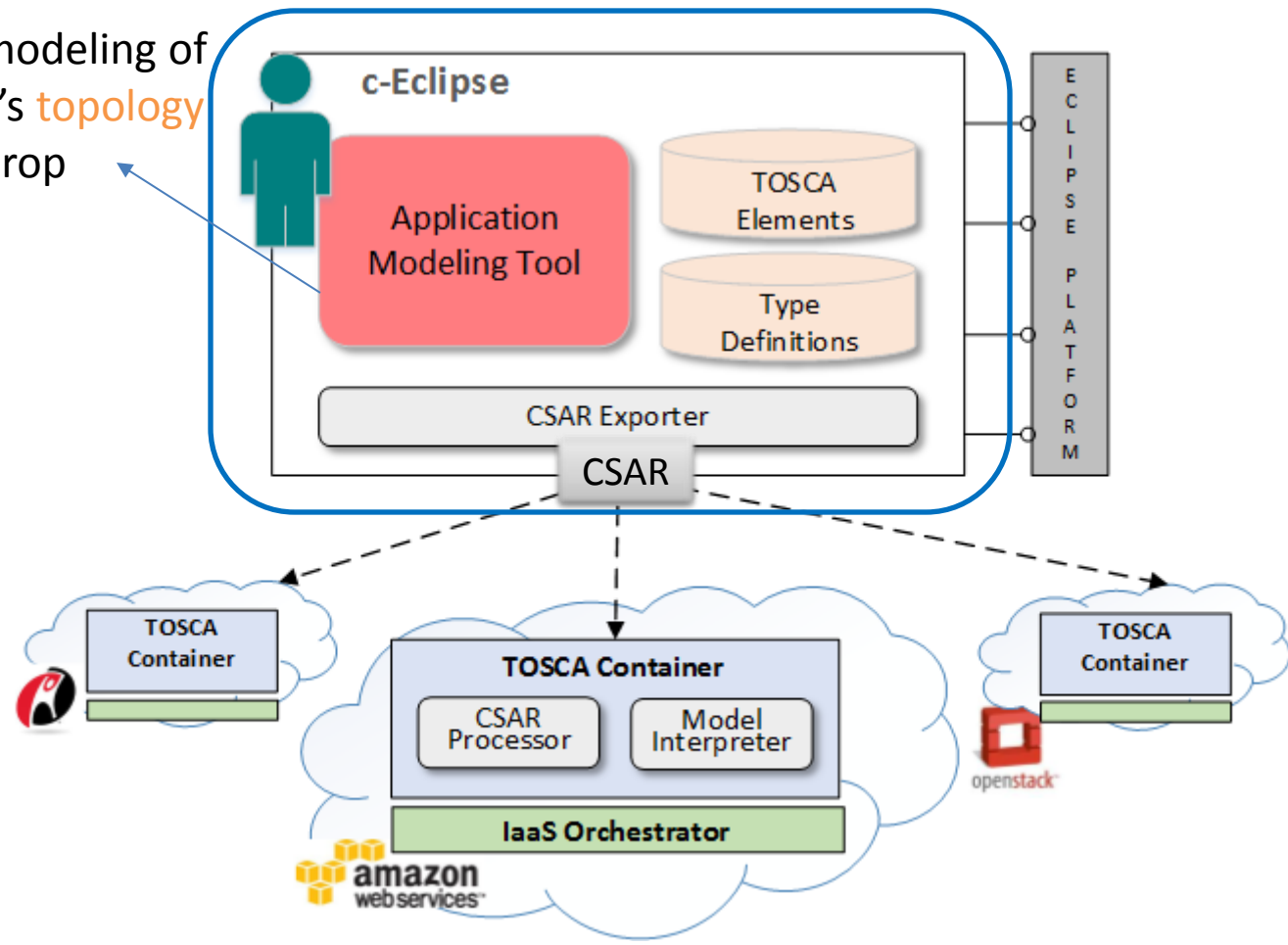


CSAR Format

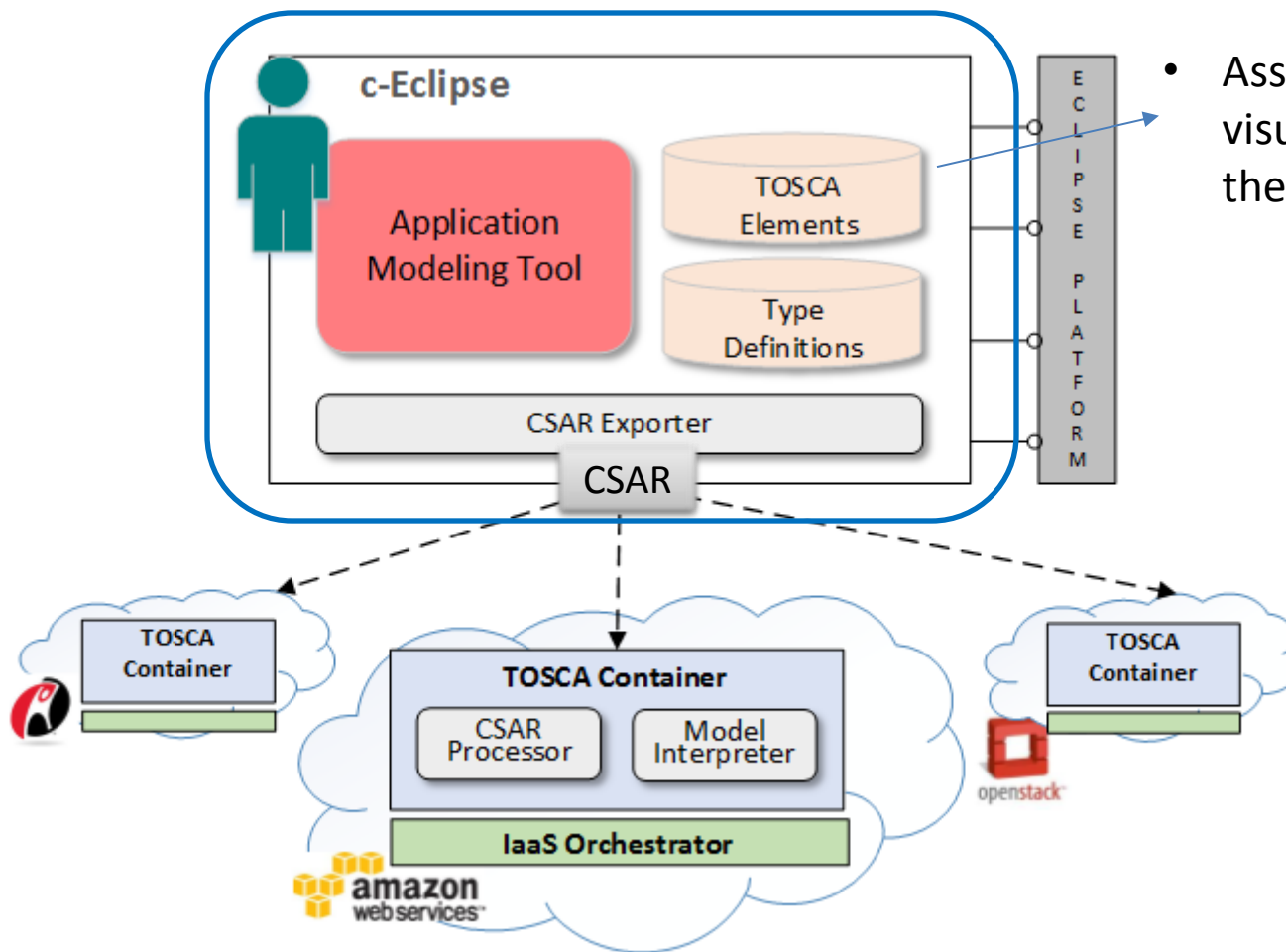
c-Eclipse Architecture

c-Eclipse Architecture

- Graphical modeling of application's topology
- Drag-and-drop interface

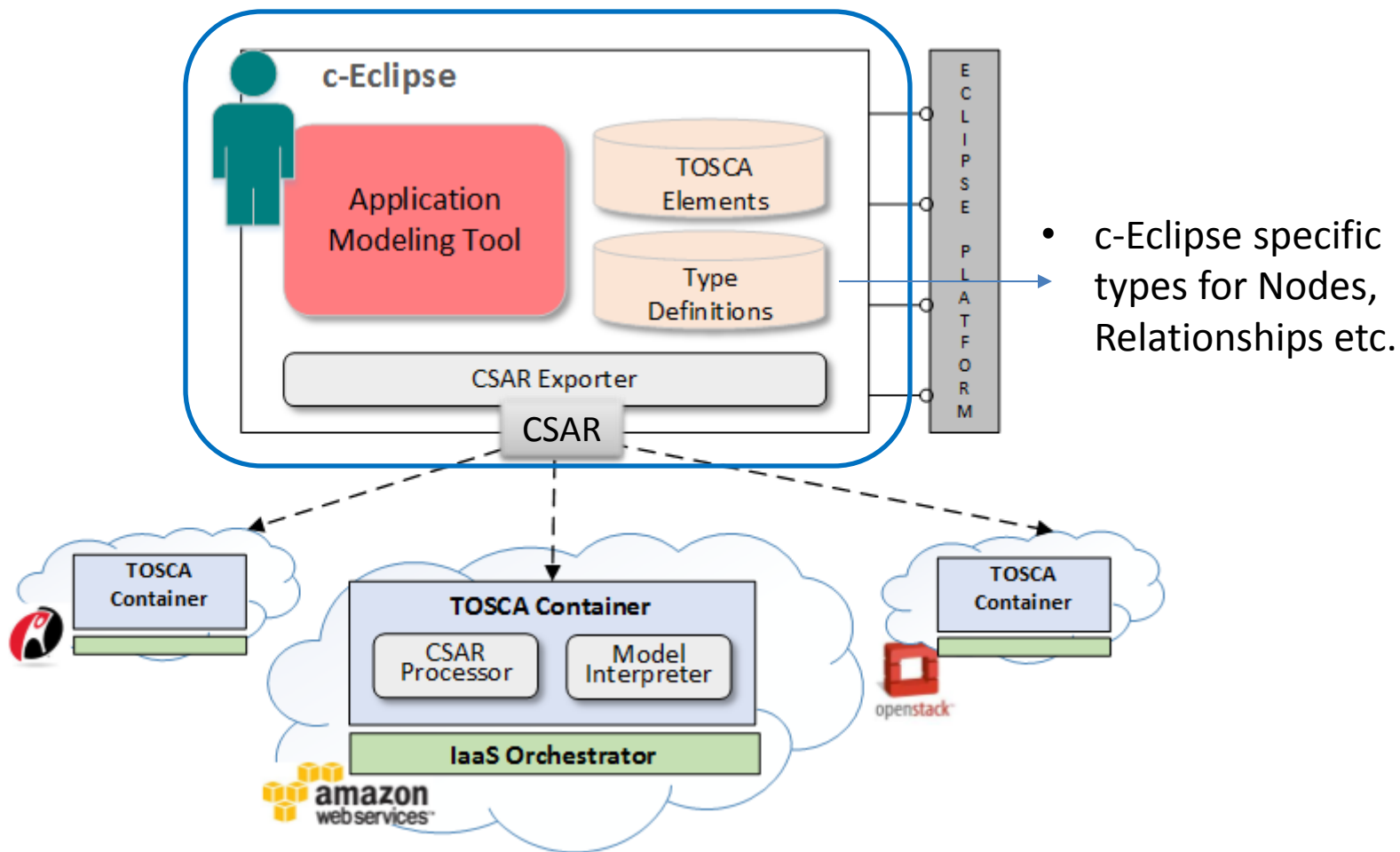


c-Eclipse Architecture

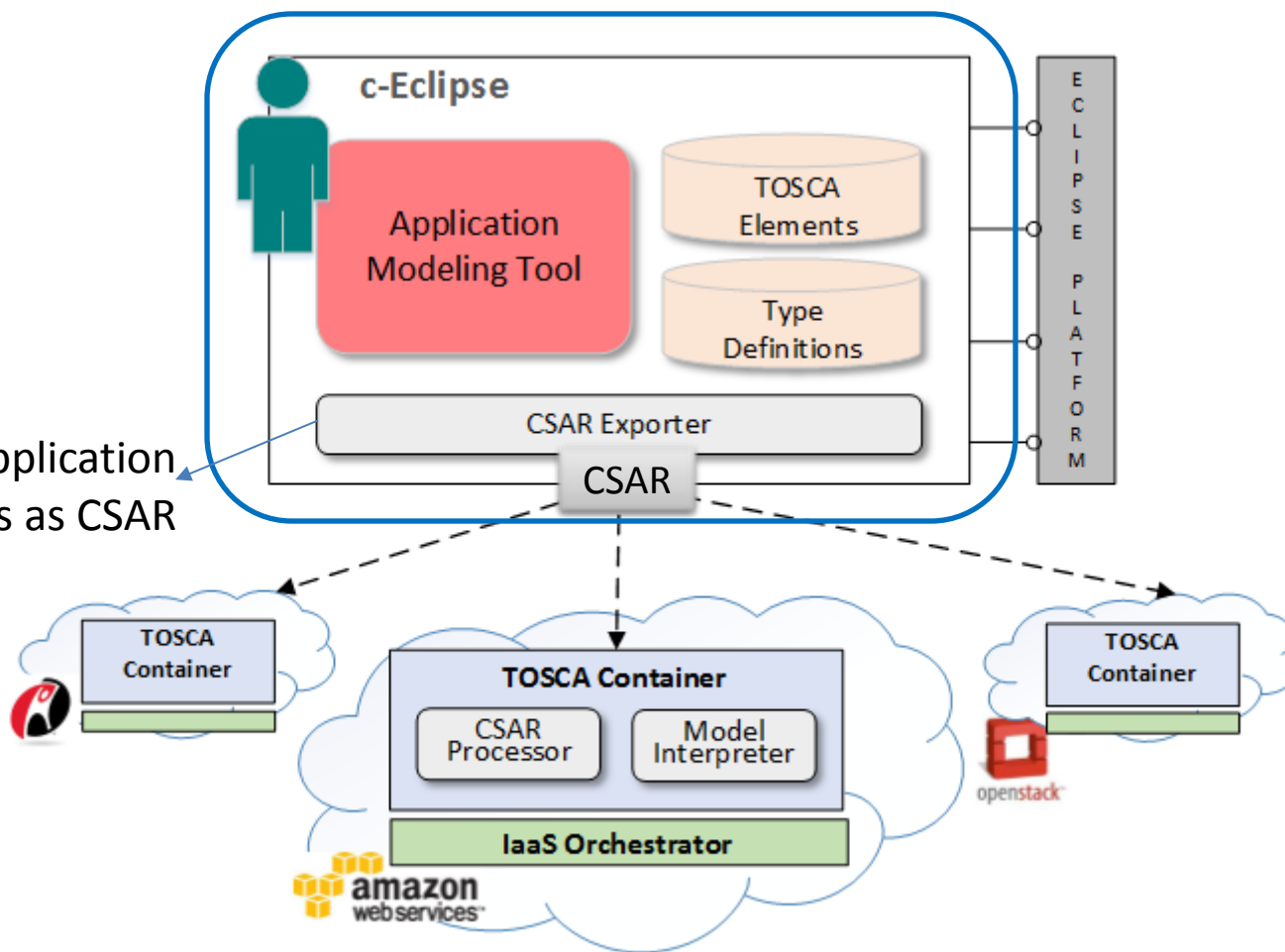


- Associated with visual elements by the Modeling Tool

c-Eclipse Architecture

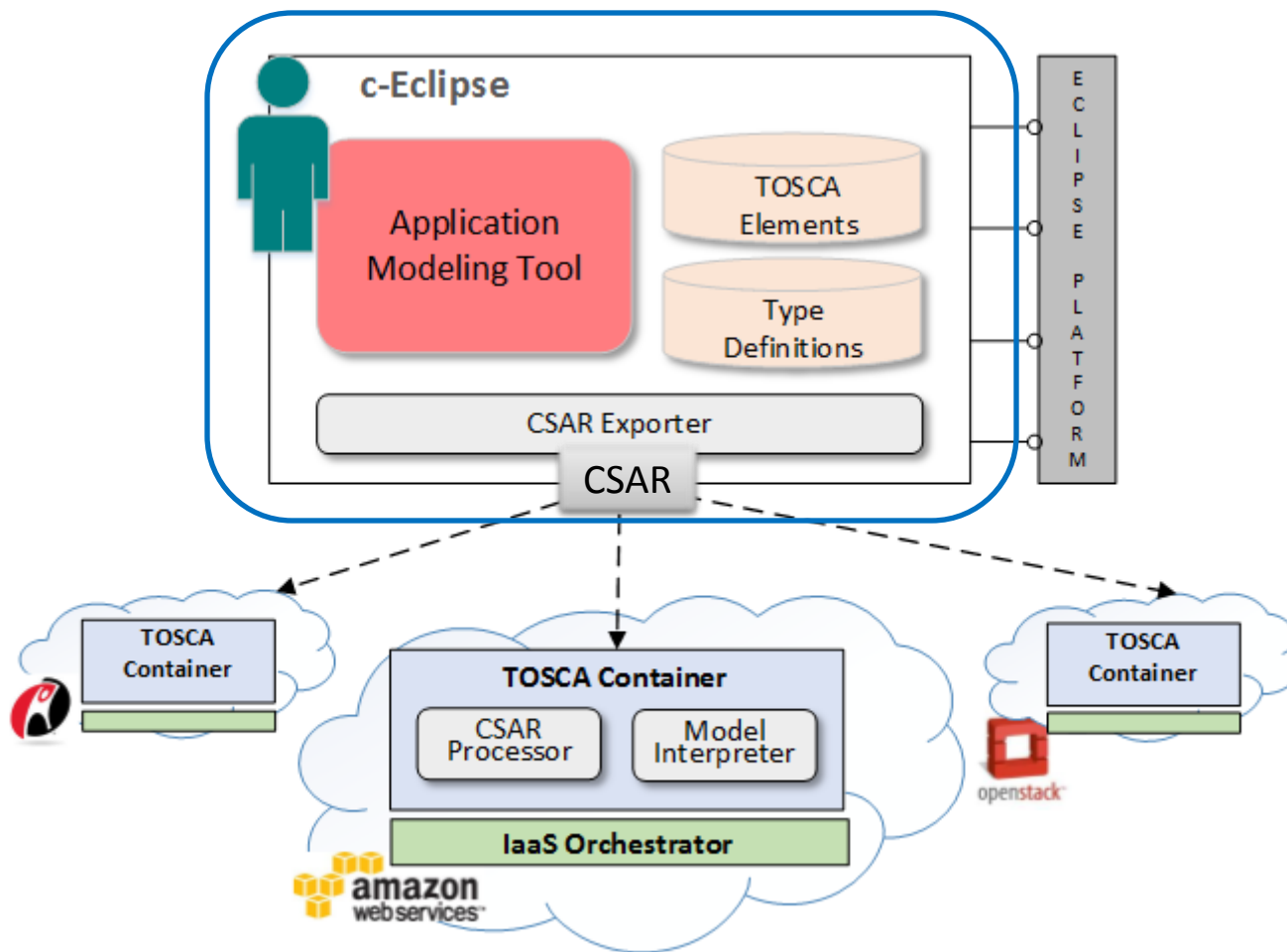


c-Eclipse Architecture



- Packages application descriptions as CSAR archives

c-Eclipse Architecture



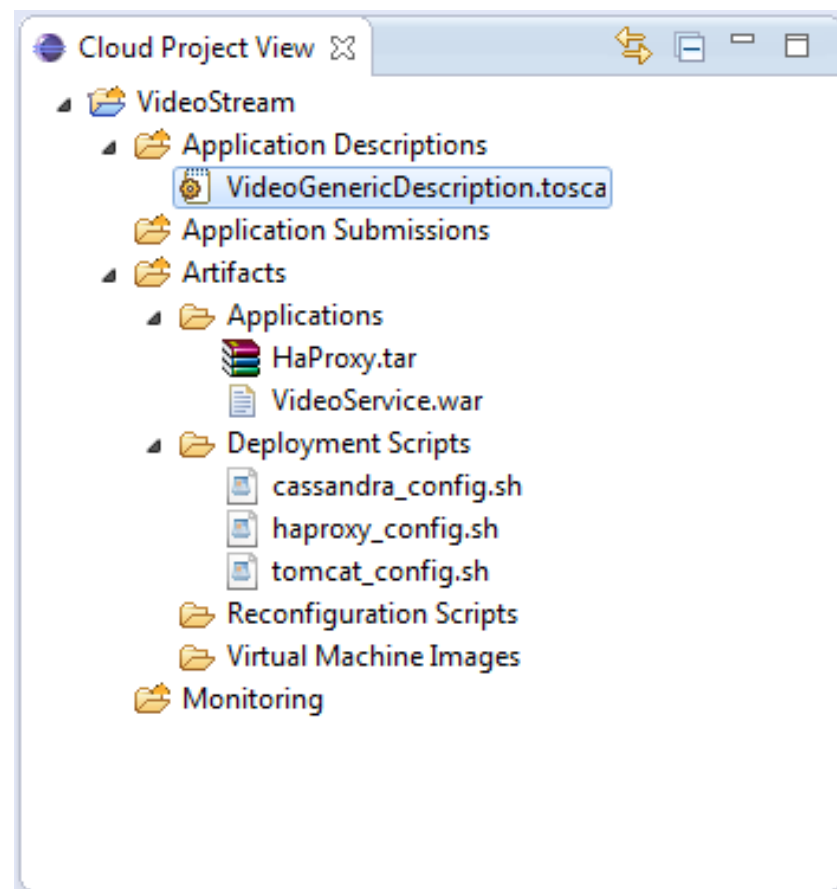
c-Eclipse Features

Application Modeling Tool

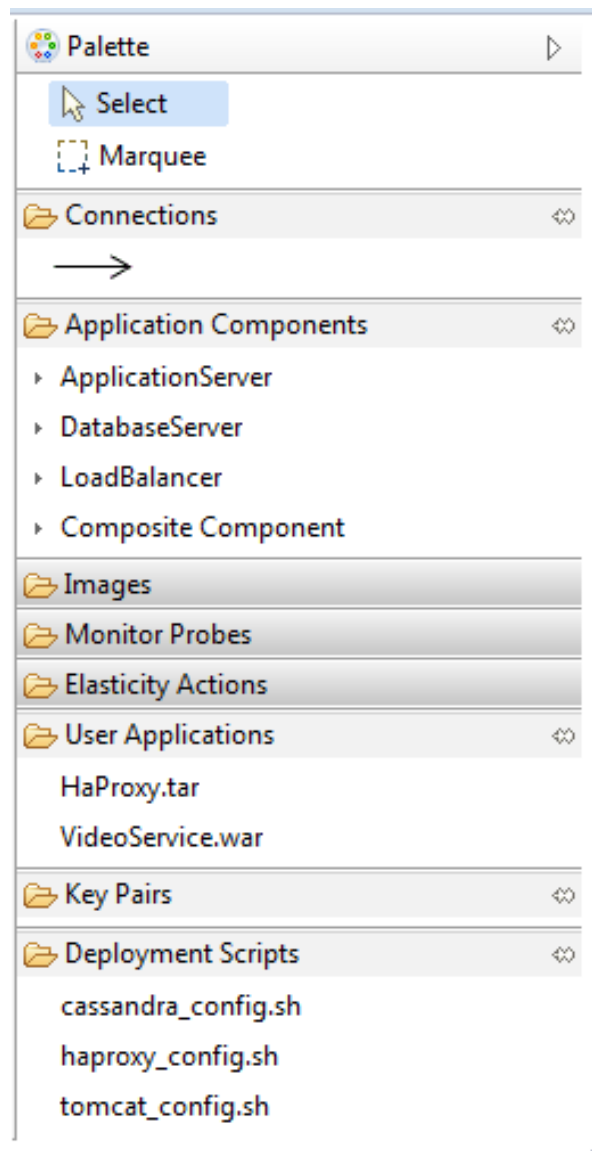
The screenshot displays the Application Modeling Tool interface. On the left, the **Cloud Project View** shows a tree structure for a project named 'VideoStream', including folders for 'Application Descriptions', 'Application Submissions', 'Artifacts', and 'Monitoring'. The main area is the **Canvas**, which contains three application components: 'Load_Balancer' (containing 'HaProxy.tar' and 'haproxy_config.sh'), 'Application_Server' (containing 'VideoService.war' and 'tomcat_config.sh'), and 'NoSQL_Database' (containing 'cassandra_config.sh'). An arrow points from the Application_Server to the NoSQL_Database. On the right, the **Palette** lists various components and scripts, including 'ApplicationServer', 'DatabaseServer', 'LoadBalancer', 'Composite Component', 'Images', 'Monitor Probes', 'Elasticity Actions', 'User Applications', 'Key Pairs', and 'Deployment Scripts'. At the bottom, the **Properties View** shows the configuration for the 'Application_Server' component, including fields for Name, VM Image, VM Description, and Number of Instances (Initial: 1, Min: 1, Max: 3).

Cloud Project View

- c-Eclipse organizes files in a structured hierarchy
 - Just like any other Eclipse project
- Folders are placeholders for files required throughout application's lifecycle i.e.
 - Content needed to realize a deployment (executables, configuration files, VM images etc.)
- Folders' structure is automatically created on project's creation

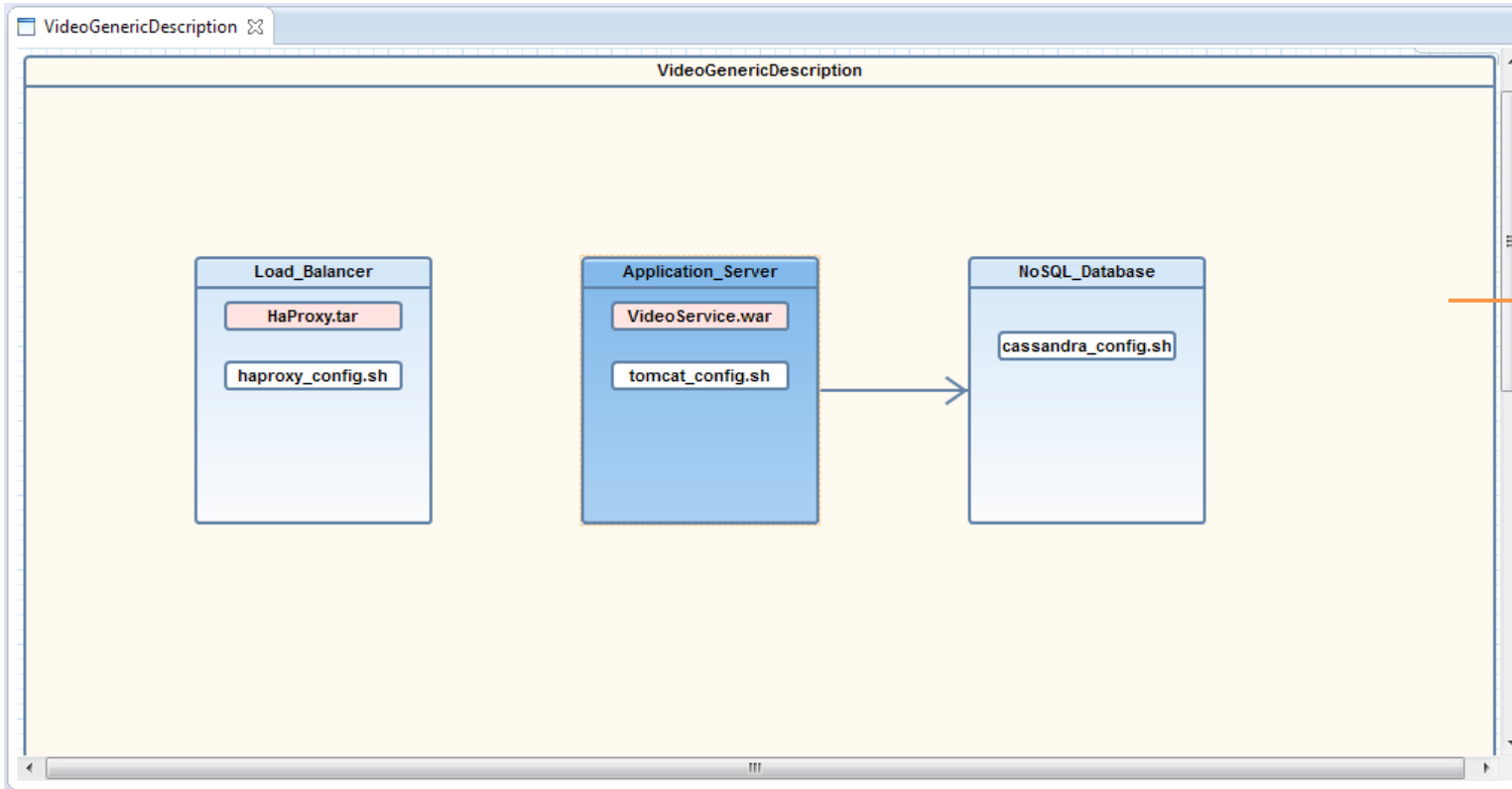


Palette



- **Connections:** Different relationship types can be specified i.e. “Depends On”, “Connects To”
- **Application Components:** Application component types + composite component
- **Images:** Provider’s images & user’s custom built images
- **Monitoring Probes:** Monitoring metrics available by the provider’s monitoring system or by the integrated to c-Eclipse monitoring system
- **Elasticity Actions:** Provider supported elasticity actions & user’s custom elasticity actions
- **User Applications:** User’s custom created applications
- **Key Pairs:** Generated by the user, used for accessing the deployed components
- **Deployment Scripts:** User’s custom configuration scripts

Canvas / Properties View

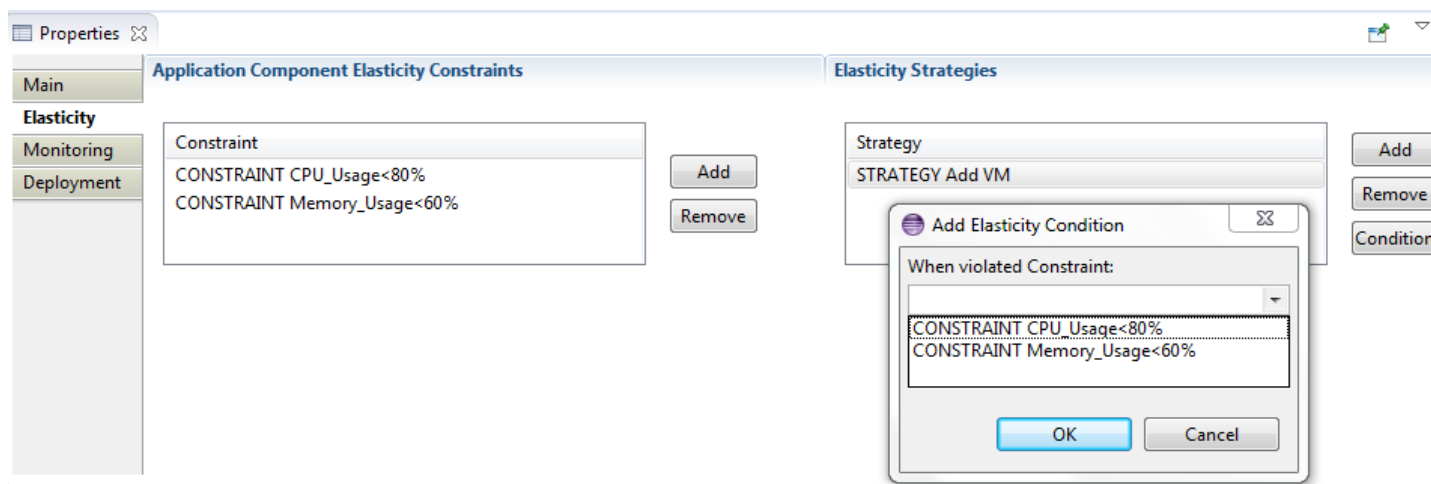


Palette elements can be dragged-and-dropped onto the canvas

More details can be specified through the Properties View

Elasticity Policies Specification

- c-Eclipse facilitates the specification of applications' elasticity policies
 - Applications can scale at runtime based on user defined policies



c-Eclipse Properties View: Elasticity Policies Specification

Elasticity Policies Specification

- *SYBL* language enables elasticity requirements description for Cloud applications
- Elasticity specification at different levels
 - Component, composite component, application
- Two types of SYBL elasticity requirements:
 - **Constraint**: “Constraint 1: CPU_Usage < 80%”
 - **Strategy**: “Strategy 1: CASE Violated (Constraint 1) : Scale_Out”

"SYBL: an Extensible Language for Controlling Elasticity in Cloud Applications", G. Copil, D. Moldovan, H. Truong and S. Dustdar, 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2013), 2013

Elasticity Policies Specification

- TOSCA v1.0 does not specify directly how to describe elasticity requirements
- c-Eclipse uses TOSCA extensibility mechanism
 - Injects SYBL elasticity directives into TOSCA

TOSCA policies express non-functional behavior or quality-of-services for an application

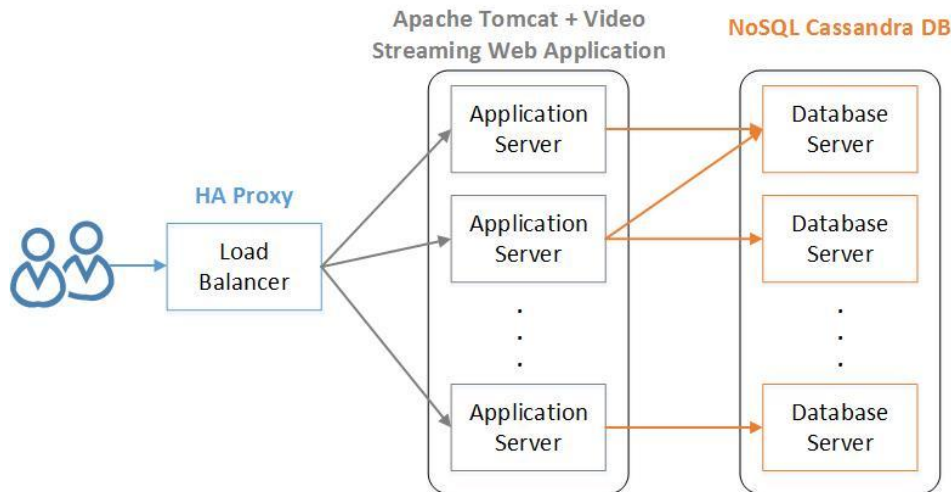
```

<tosca:Policy type="ElasticityStrategy"
  id="CPU_Strategy">
  <tosca:Properties>
    <ElasticityStrategyProperties>
      <Condition>
        <BinaryRestriction Type="GreaterThan">
          <LeftHandSide>
            <Metric>CPU_Usage</Metric>
          </LeftHandSide>
          <RightHandSide>
            <Number>80</Number>
          </RightHandSide>
        </BinaryRestriction>
      </Condition>
      <ToEnforce ActionName="Add_VM" />
    </ElasticityStrategyProperties>
  </tosca:Properties>
</tosca:PolicyTemplate>
    
```

c-Eclipse in Action

Use-Case Scenario

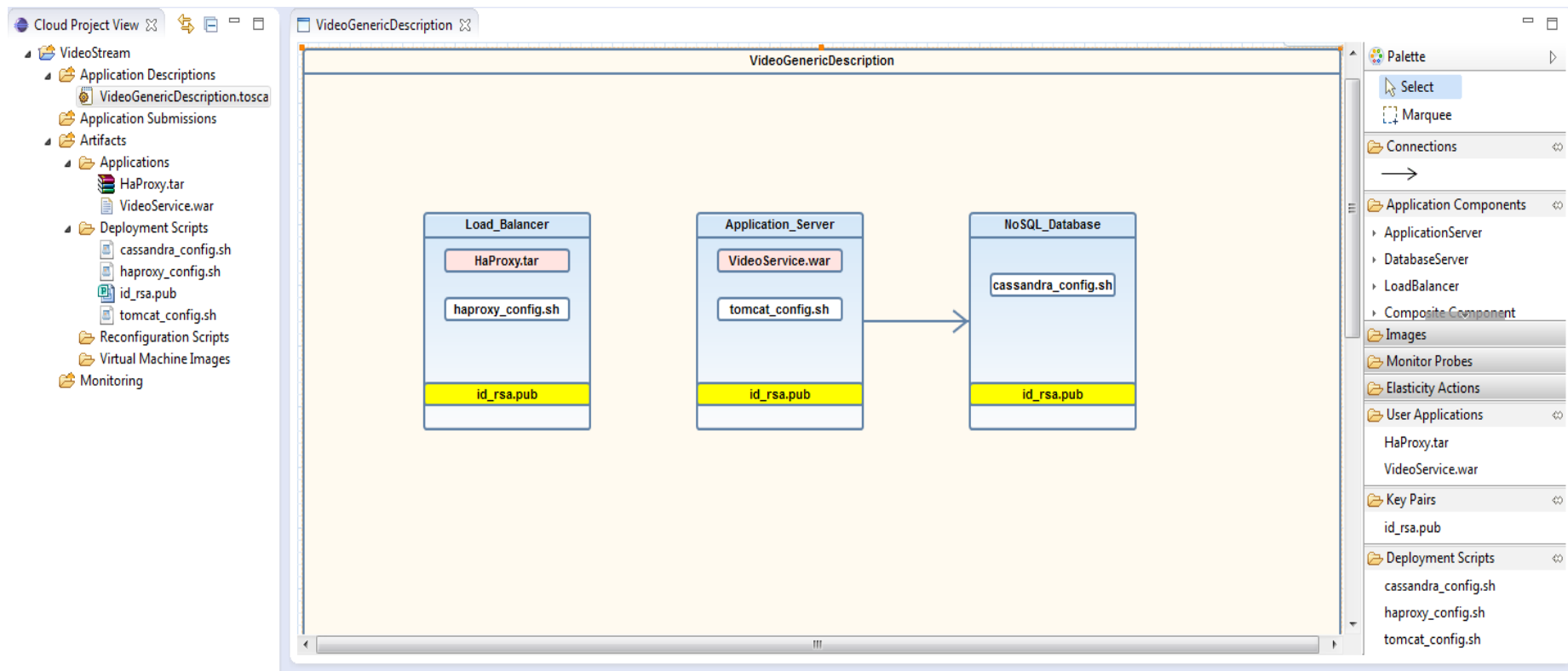
- 3-Tier application
 - Video Streaming Service to Online Users



- Deploy application over
 - Amazon EC2 infrastructure
 - OpenStack-compliant infrastructure

Generic Application Description

- No Cloud provider selected yet
- Only user-defined files are used i.e. software files, configuration scripts etc.



Application Description

Customization: Amazon EC2

- Palette is populated with vendor-specific information (Using vendor's API)
- Authentication Token View gives an overview of user's credentials

The screenshot displays a 3-Tier Video Stream Service architecture. The components are:

- Load Balancer:** Contains `HaProxy.tar` and `haproxy_config.sh`. It is associated with `aws_id_rsa.pub` and `ami-3007f247`.
- Application Server:** Contains `VideoService.war` and `tomcat_config.sh`. It is associated with `aws_id_rsa.pub` and `ami-3007f247`.
- NoSQL Database:** Contains `cassandra_config.sh`. It is associated with `aws_id_rsa.pub` and `ami-6f310c1b`.

The Palette sidebar on the right lists various resources:

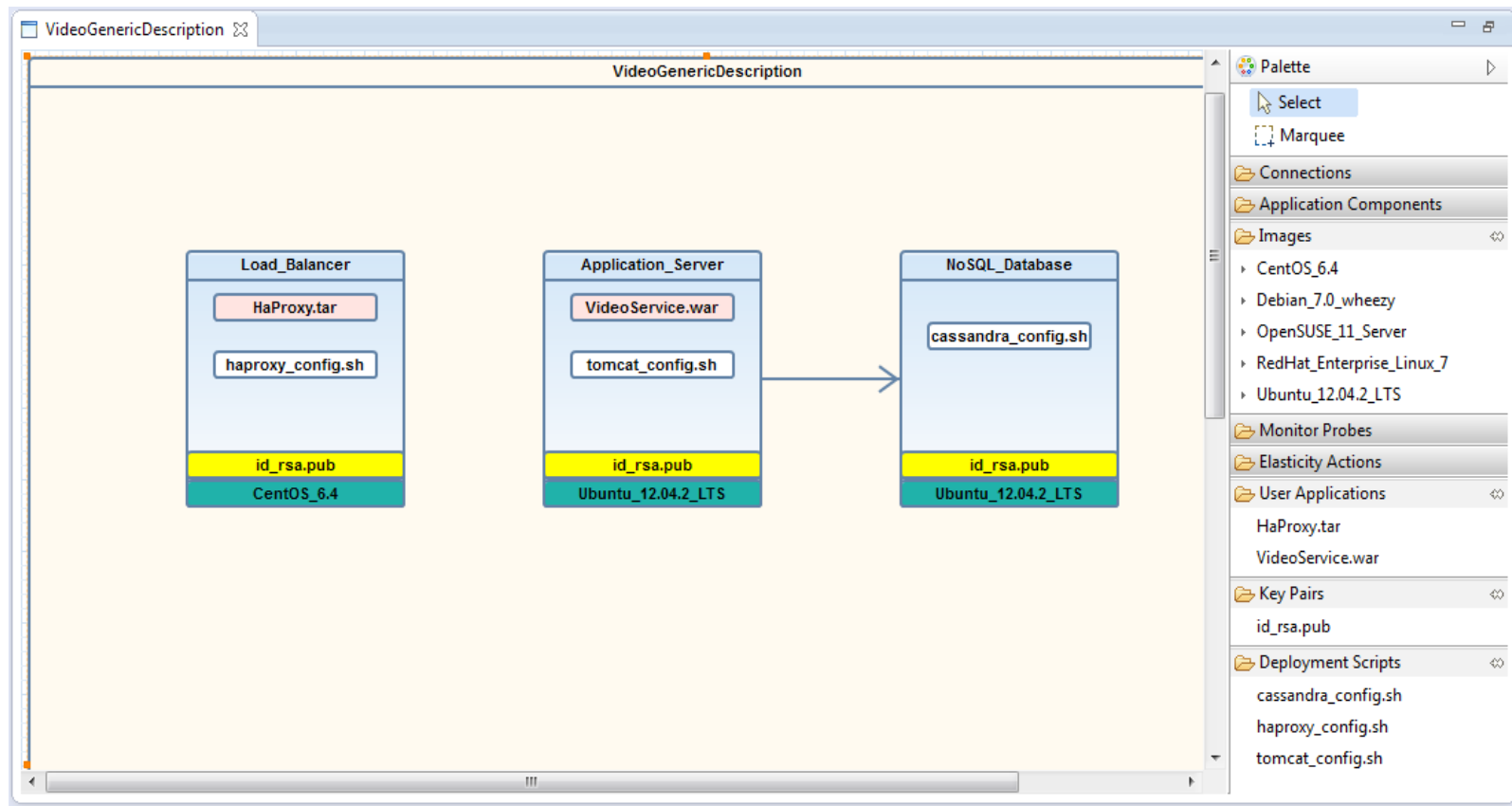
- EC2 images:** `ami-3007f247`, `ami-6d310c19`, `ami-6f310c1b`, `ami-e5fbce91`, `ami-ffffbce8b`.
- CloudWatch metrics:** `CPUUtilization`, `DiskReadBytes`, `StatusCheckFailed_System`, `StatusCheckFailed`.

The Authentication Token UI table at the bottom shows:

ID	Type	State	Time Left
<input checked="" type="checkbox"/> AWS Authentication Token # 1 @ 181B'S Authentication To		Active	infinite

Application Description

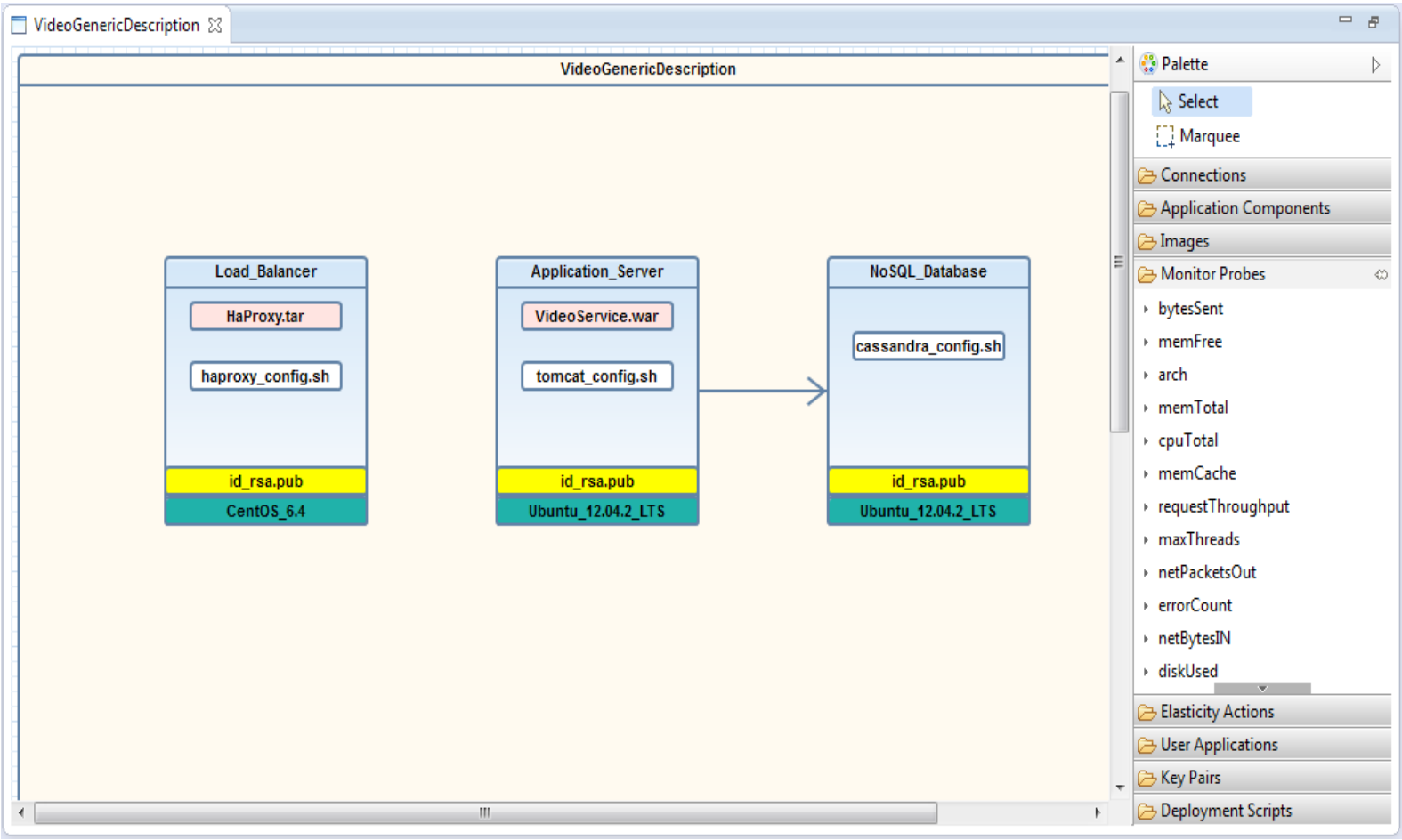
Customization: OpenStack-compliant



OpenStack infrastructure images



Application Description

Customization: OpenStack-compliant



JCatascopia monitoring probes

TOSCA Container Implementation

- Implemented two *prototypical TOSCA Containers* for
 - Amazon EC2 
 - Amazon API
 - around 450 LOC
 - OpenStack-compliant infrastructure 
 - jClouds
 - around 600 LOC
- Deployed the two containers on the respective infrastructures
 - Ready to receive application deployment requests

Deployment over EC2 & OpenStack

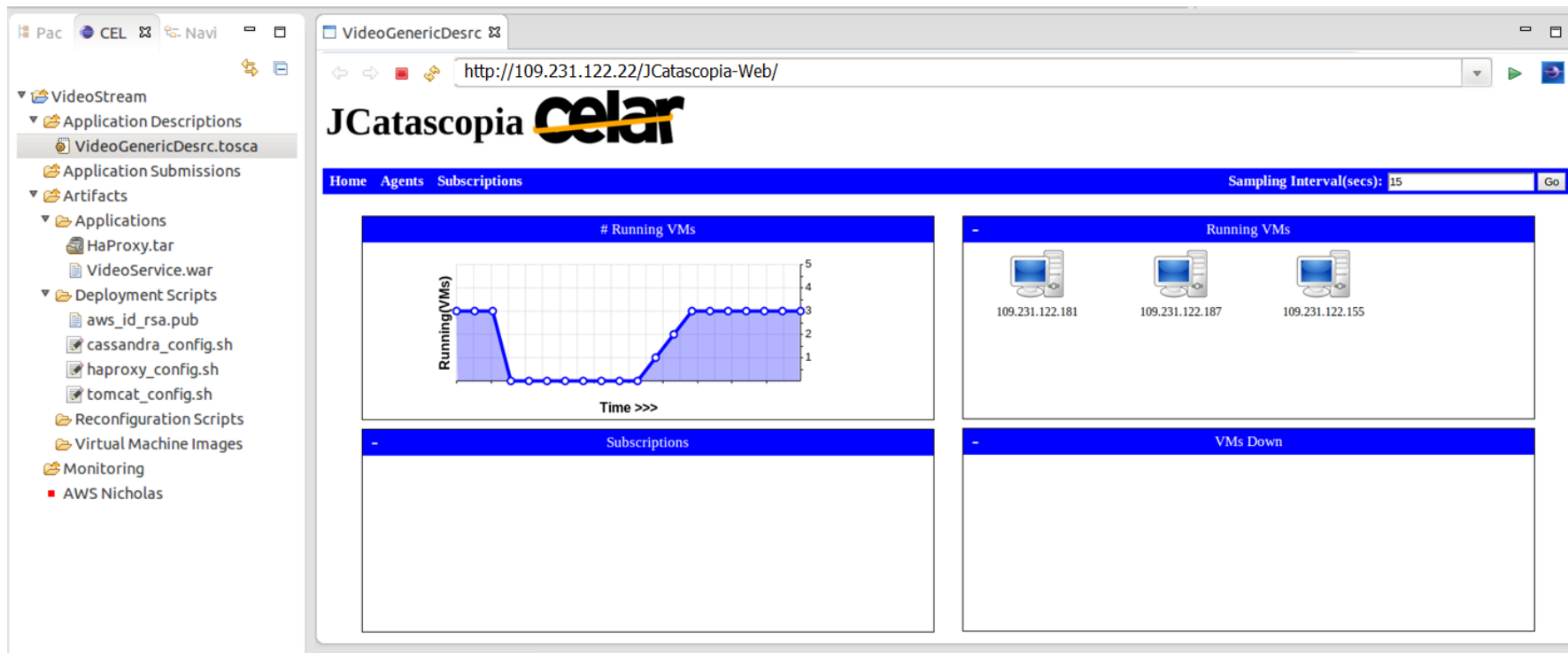
- Applications' deployment request are sent
- The two applications are up and running on the 2 infrastructures
- The status of the two deployments is shown in the *Application Deployments View* of c-Eclipse

Application Name	Status	Instance ID	IP Address
3-Tier Video Stream Service (3)	DEPLOYED		
Load Balancer	RUNNING	i-13461e53	172.31.43.237
Application Server	RUNNING	i-aa441cea	172.31.31.71
NoSQL Database	RUNNING	i-ab441ceb	172.31.37.226
3-Tier Video Stream Service (3)	DEPLOYED		
Load Balancer	RUNNING	8e3c4cb6	10.16.5.3
Application Server	RUNNING	fd9f7af2a3c2	10.16.5.4
NoSQL Database	RUNNING	21d9f7af2a4c1	10.16.5.5

c-Eclipse Application Deployments View

Monitoring Application Deployments

- OpenStack infrastructure does not provide monitoring system
 - We integrated the JCatascopia open-source monitoring system to c-Eclipse



"JCatascopia: Monitoring Elastically Adaptive Applications in the Cloud", D. Trihinas and G. Pallis and M. D. Dikaiakos, 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014), 2014

Conclusions

Conclusions

- ✓ c-Eclipse built on top of the Eclipse Platform as an Eclipse plug-in
 - ✓ Project proposal to Eclipse Foundation is currently under evaluation
- ✓ Adopts open Cloud standards (TOSCA) for promoting applications' portability
- ✓ Use of SYBL language to enable the description of applications' elasticity requirements'
- ✓ Integration with open-source monitoring system
- ✓ Successfully tested on public & private Clouds

Acknowledgements



www.celarcloud.eu



co-funded by the
European Commission

c-Eclipse

Code: <https://github.com/CELAR/c-Eclipse>

Demo Video: <http://grid.ucy.ac.cy/CELAR/icwe2014/>



LINC

Laboratory for Internet Computing
Department of Computer Science
University of Cyprus

<http://linc.ucy.ac.cy>