1.1 Get	ting Started	2
1.1.1	Open CAMF Perspective	2
1.1.2	Create a new Cloud Project	2
1.1.3	Manage Cloud Providers	4
1.1.3.1	Add Cloud Provider	4
1.1.3.2	2 Edit Cloud Provider	5
1.1.4	Cloud Application Description	6
1.1.4.1	Create application description file	6
1.1.4.2	2 Fetch Cloud provider info	7
1.1.4.3	3 Check authentication details	8
1.1.4.4	Describe Cloud application structure/contextualization	8
1.1.4.5	5 Specify application's elasticity requirements1	1
1.1.5	Submit application description to Cloud provider1	2
1.1.6	Deploy application to Cloud provider1	4
1.1.6.1	L Create deployment file1	4
1.1.6.2	2 Watch deployment status1	5
1.1.7	Monitoring1	5
1.1.7.1	Create JCatascopia Monitoring Probe1	6

1.1 Getting Started

1.1.1 Open CAMF Perspective

The CAMF Perspective contains the Views and Editors utilized by CAMF.

Open the CAMF Perspective: Window -> Open Perspective -> Other -> CAMF. (Figure 1)

Open Perspective	
Image: CAME Image: Debug Image: Graphiti Image: Java Image: Java Browsing Java Type Hierarchy Image: Ja	
ОК	Cancel

Figure 1

You can close the *Cloud Information Tool* view, since we are not going to need it for the current tutorial.

1.1.2 Create a new Cloud Project

Right Click in the Cloud Project View -> New -> Other -> Cloud Application Management Framework -> Cloud Project. (Figure 2)

New	
Select a wizard Create a new CELAR Cloud project	
Wizards:	
type filter text	
 Cloud Application Management Framework Application Description Cloud Project Java Java JavaScript Plug-in Development Server Web XML Other Examples 	
(?) < Back Next > F	inish Cancel

Next -> Give a name for the newly created project -> Next -> Select a Cloud Provider to associate the project with (Figure 3) -> Finish (if you have not already added providers to your workspace you can do so by clicking on the Edit Cloud Providers button and following the instructions in section 1.1.3).

New CELAR Cloud Project	
Cloud Provider Selection Page Specify the Cloud Provider that should be used	
Available Cloud Providers	
OpenStack_Nephelae	Edit Cloud Providers
Cloud Provider Info	
OpenStack_Ivepnelae Type: OpenStack Cloud Provider	Ŧ
4	4
? K Sack Next >	Finish Cancel

Figure 3

The new project is created and the project's folder structure is shown in the Cloud Project View (Figure 4). The created folders will be used as follows:

- Application Descriptions: holding topology blueprints defined using TOSCA. Each blueprint can be unique by specifying a different structure and management operations for the application at hand.
- Application Deployments: holding IaaS-flavored blueprints along with important operational records (past and current) regarding different deployment. Amongst other, these can include date/time, the target deployment IaaS, version of the application description, operational costs, etc.
- Artifacts: holding concrete software implementations required for the successful deployment and correct operation of the application. These include, and not limited to: executable and/or 3rd party libraries, custom virtual machine images, Chef cookbooks, O/S-specific configuration scripts, etc. The Cloud project structure, allows for artifacts to be referenced by multiple descriptions thus avoiding unwanted duplication.
- Monitoring: holding monitoring metric collectors prepared by the developer. These probes can be placed anywhere on the application stack ranging from the virtualization layer upwards to the application itself, in order to obtain meaningful metrics that report the runtime health and performance of the application.



Figure 4

1.1.3 Manage Cloud Providers

1.1.3.1 Add Cloud Provider

If the Cloud Providers wizard is not open go to Window -> Preferences -> Cloud Application Management Framework -> Cloud Providers

Add -> OpenStack Cloud Provider (Figure 5) -> Next: Give a name for the OpenStack compliant provider to which you have already been registered and have acquired the necessary credentials (username and password). Fill in the Access Id (username) and Endpoint (URI) and click Finish -> OK. Here you can specify as many providers as you want. The selection of a provider to submit an application for deployment is done at a later stage.



1.1.3.2 Edit Cloud Provider

If the Cloud Providers wizard is not open go to Window -> Preferences -> Cloud Application Management Framework -> Cloud Providers

Select the provider to be edited -> Edit -> OpenStack Cloud Provider -> Next: You can update the Access Id (username) and/or Endpoint (URI) (Figure 6). Click Finish -> OK.

Preferences		
type filter text	Cloud Providers	() ▼ () ▼ ▼
 ▷ General ▷ Ant ▲ Cloud Application Manage Cloud Providers Graphiti Test Preferences ▷ Help ▷ Install/Update ▷ Install 	Add, remove or edit Cloud Provider descriptions. The checked Cloud Provider is taken as the default Cloud Provider when creating new c-Eclipse projects. Known Cloud Providers: Name Type OpenStack OpenStack Cl	Add Import
 JavaScript Plug-in Development Run/Debug Server Team Validation Web XML 	OpenStack Cloud Provider Specify the attributes of your OpenStack Cloud Provider Cloud Provider Settings Cloud Provider Name: OpenStack Access ID: camf:stalosof OpenStack Endpoint: http://10.16.3.3:5000/v2.0	Remove
	Services Service URL	
< F	ОК	Cancel
	Cancel	

1.1.4 Cloud Application Description

1.1.4.1 Create application description file

Right Click on a Project in the Cloud Project View -> New -> Other -> Cloud Application Management Framework -> Application Description -> Next -> Give a name for the newly created description -> Finish.

The application description file is created under the Application Descriptions folder with extension .tosca (Figure 7).





1.1.4.2 Fetch Cloud provider info

When creating the first application description in a project, the framework tries to fetch the following data from the specified OpenStack Cloud provider:

- Images: Virtual machine images that are made available through the OpenStack Image Service
- *Networks:* Virtual networks to enable compute servers to interact with each other and with the public network
- Key Pairs: User-created keys (i.e. rsa keys) that can be used to access the instances once they have been launched
- *Flavors*: A flavor represents a set of virtual resources. Flavors define how many virtual CPUs an instance has and the amount of RAM and size of its ephemeral disks. OpenStack provides a number of predefined flavors that you can edit or add to. Users must select from the set of available flavors defined on their cloud.

A pop up window asks if you want to fetch this data (Figure 8).





Click Yes and add the missing credentials (password) for the selected provider -> Finish (Figure 9).

Describe and	authentication token
Provide aut	
OpenStack	Credentials
Access ID:	camf:stalosof
Endpoint:	http://10.16.3.3:5000/v2.0
Secret ID:	******



1.1.4.3 Check authentication details

Once the user has provided her full credentials (username and password) for the specified endpoint and has been authenticated by the respective Cloud provider, her authentication details are shown in the *Authentication Token UI* view (Figure 10).

🗔 Properties Authentication Token UI 💥 📩 Applicatio	n Deployments			🗂 • × o o i 🖗 = 🖬
D	Туре	State	Time Left	
OpenStack Authentication Token # 1 @ camf:stalosof	OpenStack Authentica	Active	infinite	



1.1.4.4 Describe Cloud application structure/contextualization

Right click on the application description file -> Open With -> Tosca Diagram Editor. You can also open the xml TOSCA file at any time, by selecting Open With -> Text Editor.

All the information required for the description can be found in the *Palette* (Figure 11) and/or in the *Properties* view (Figure 12). The Palette includes some generic elements (application components and relationships), as well as Cloud provider specific elements previously fetched (images, networks and key pairs). Elements from the Palette can be simply dragged and dropped onto the center *Canvas* of the tool.

😳 Palette 🗘	>
ि Select	
[]] Marquee	
➢ Connections	
Application Components	
🔁 Images 🔹	¢
regionOne/0133e5a7- a27c-4cc3-9c78-6b6127b4a440	
regionOne/eec3d5c9-75d0-4a95-a66b- b08f9a963bc0	
regionOne/65b62ef8-f51f-4eac-95e5- fd3a473c7efd	
regionOne/83287548-1666-49de-bf35-	
regionOne/2c02fcae-1fdd-4321- a734-90f7f57997b8	
> Networks	¢
> 3fa16890-06a7-44b5-a8e1-e2cfb8d28507	
> 8ebb464c-ad94-464b-ab87-28cfc46d9ecb	,
🗁 Monitor Probes	
➢ Elasticity Actions	
🔁 User Applications	
😕 Key Pairs	co
rsa_key.pub	
🔁 Deployment Scripts	

Figure 11

The rest of the information (flavors) can be found in the *Properties* view (Figure 12).

Main	Application Co	monneat Pronecties	1
Basticity	Name	linc-VM	1
Deta/Load	(valine)	m1.small	
Monitoring	VMImage	mI.medium	Ime
Parameters		scan-flavor	-
Deployment	VM Detion:	JC-Flavor	
	VM Type:	3	र्ग ।



Users can also use image and configuration files stored in their local file system, simply by importing them into the respective folders of a Cloud Project (Virtual Machine Images, Deployment Scripts folders). Once the files are imported into the Cloud Project, they are also added in the Palette.

Right Click -> Import -> General -> File System -> Next -> Browse -> Select a directory to import from. (Figure 13)

the first Newsyme Associate Property Run-	Window Mate	
	10.0	Qualt Arrens 🔡 🔁 Ressarts 🗰 Cald
Coul Project View 11	mydeplamorDecaption II	* 0
a D MyApplication	(WyAquificationDescription	C B Peiete
Application Decorption Application Decorption Application Solution Application Solution Application Addect Addect Addect Addect Provide Solution Provide		Compose Composed Comp
	(2) the Net Continue	P Address

Firstly decide the number of application components to put in the description. Each application component corresponds to a virtual machine instance. Then, for each application component the image (green box), flavor and network(pink box) must be specified. Additional attributes can be specified, such as key pair (yellow box), deployment/configuration scripts (beige box), and initial number of instances (Figure 14).

			myApplica	tionDescription		
	App_Comp config_webs Sta16890-06a7-44 rsa_key; regionDisc@1334	oment app.sh bb5-a8et pub e6e7.a27				
-						
Properties	11 💝 Authentic	cation Token UI 🗂 Application	m Deployments		ā	
e (Properties Main	II 😚 Authentic	ration Token UI 🗂 Application mponent Properties	m Deployments	Numbe	r of Instances	
Properties Main Elasticity	33 🥵 Authentic Application Co	cation Token U. C Application mponent Properties	n Deployments	Numbe	r of Instances	,
Properties Main Elesticity Data/Load	II 😚 Authentic Application Co Name:	ation Token UI — Application mponent Properties App_Component	P Deployments	Numbe	r of Instances	, 1
Properties Properties Main Elasticity Data/Load Monitoring	 S S Authentia Application Con Name: VM Image: 	cation Token UI C Application mponent Properties App_Component regionOne;0133e5s7-a27c-4cc	P Deployments Add Image	Numbe Initial: Min:	r of Instances	
Properties Main Elesticity Data/Load Monitoring Parameters	 Authentia Application Con Name: VM Image: 	cation Token UI C Application mponent Properties App_Component regionOne/0133e5a7-a27c-4cc	P Deployments Add Image	Numbe Initial: Min:	r of Instances	
Properties Main Elasticity Data/Load Monitoring Parameters Deployment	II S Authentic Application Co Name: VM Image: VM Detion:	cation Token UI C Application mponent Properties App_Component regionOne/0133e5a7-a27c-4cc	P Deployments Add Image	Numbe Initial: Min: Mai:	r of Instances	

Figure 14

1.1.4.5 Specify application's elasticity requirements

You can specify elasticity policies for your application so that it can scale at runtime based on the defined policies.

There are two different types of elasticity policies:

- *Elasticity Constraint:* Used to express the constraints of an application, related to cost, performance and other application-quality metrics. Here the application user does not specify the exact actions to be enforced when a constraint is violated. Instead, the appropriate actions are determined by the underlying intelligent elastic Resource Provisioning System¹.
- *Elasticity Strategy:* Used to express specific strategies that should be enforced by the execution environment when specific constraints are violated.

The policy presented in Figure 15 describes an elasticity constraint where the application provider wants the CPU usage of a database cluster to be less than 80%. In Figure 15 the application provider uses an elasticity strategy to specify that when the CPU usage of the cluster exceeds 80%, a new database VM should be added to the cluster.

Main	Application Component Elasticity Constraints		Elasticity Strategies	
Elasticity Monitoring Deployment	Constraint CONSTRAINT CPU_Usage<80%	Add	Strategy STRATEGY Add VM	[/
	CONSTRAINT Memory_Usage<60%	Remove	Add Elasticity Condition Men violated Constraint: CONSTRAINT CPU Usage<80% CONSTRAINT Memory_Usage<60% OK Cancel	Con

Figure 15

In order for the user to specify elasticity policies related to a monitoring metric, a corresponding monitoring probe (i.e. CPU Usage probe) from the Palette (Figure 16) must be assigned to the respective component by dragging the probe from the Palette and dropping it to the component.

¹ http://www.celarcloud.eu/

😳 Palatte	0
13 Select	
Marquee	
Convections	
Application Components	
Images	
Monitor Probes	-
- HAProxy	
proryBytesOUT	
proxyBytedR4	
errorflate	
servers	
requestRate	
avgilasponsaTima	
servers	
- Memory	
 Network 	
• CPU	
Elasticity Actions	
User Applications	
Key Pairs	
> Deployment Scripts	



1.1.5 Submit application description to Cloud provider

Once the application description is completed Right Click on the .tosca file under the Application Descriptions folder -> Application Submission (Figure 17).





Give a name for the submission file (Figure 18).

Deploy Application	
Submission	
Select Application to Submit	
Enter or select the parent folder:	
MyApplication/Application Descriptions	
MyApplication settings settings Application Descriptiona Application Submissions Arbitacts Monitoring	
File name: Advanced >> .	

Figure 18

Select the provider to submit the file to (Figure 19).

• Ashed all and	And the second sec
Cloud Provider Selection Select the target Cloud Provider for the App	rication
Available Cloud Providers	
ØpenStack_Nephelae	Edit Gloud Providers
Cloud Provider Info	
OpenStack_Nephelae Type: OpenStack Cloud Provider	
e .	
(?) <gech !<="" td=""><td>Sot> Enish Cancel</td></gech>	Sot> Enish Cancel



The submission file is created under the Application Submission folder (Figure 20).





1.1.6 Deploy application to Cloud provider

1.1.6.1 Create deployment file

What remains is to send the deployment request to Cloud provider of your choice. Right Click on the .tosca file under the Application Submissions folder -> Application Deployment (Figure 21).



Figure 21

Select the Cloud provider over which you wish to deploy your application -> Finish (Figure 22).

E Deploy Application			
Select Cloud Provider			
Select Cloud Provider for Ap	plication Deploymen	it	
Cloud Provider			
*			
(?) < B	ack Next >	Finish	Cancel



1.1.6.2 Watch deployment status

After deployment, the application developer can interact with the *Deployment* view (Figure 23), so as to instantly retrieve the applications' operational status without leaving CAMF's workspace. The *Deployment* view provides a snapshot of all application deployments grouped per target IaaS. Each deployment is accompanied with provider-specific properties such as IP addresses of each component, instance IDs, uptime and cost information, if available.

pplication Name	Status	Instance ID	IP Address	
3-Tier Video Streaming Service (3)	DEPLOYED			
Load Balancer	RUNNING	i-13461e53	109.231.122.181	
Application Server	RUNINING	i-aa441cea	109.231.122.187	
NoSQL Database	RUNNING	i-ab441ceb	109.231.122.155	
3-Tier Video Streaming Service (3)	DEPLOYED			
Toad Balancer	RUNNING	8e3c4cb6	10.16.5.3	
Application Server	RUNNING	fd9f7af2a3c2	10.16.5.4	
NoSQL Database	RUNNING	21.d9f7af2a4c1	10.16.5.5	



1.1.7 Monitoring

CAMF enables integration with different monitoring systems, enabling its users to collect performance metrics regarding any deployed application without being required to deal with external software environments.

Currently, CAMF is fully integrated with JCatascopia², an automated, multi-layer interoperable monitoring system for elastic Cloud environments. Besides the standard probe suite available in JCatascopia, CAMF allows application developers to define and implement custom metric collectors that adhere to JCatascopia's modular architecture. These probes can be placed anywhere on the application stack ranging from the virtualization layer upwards to the application itself, in order to obtain meaningful metrics that report the runtime health and performance of the application.

1.1.7.1 Create JCatascopia Monitoring Probe

Users can write custom java monitoring probes in CAMF.

m	ReationDescription Palette
	Select
	[_] Marquee
	E Connections
App_Component	😂 Application Compo
	ApplicationServer
config_webapp.sh	DatabaseServer
	LoadBalancer
21+16830-06+7-4415-48+1	Composite Compon
rsa_key,pub	Co-Images
regionOne@121e5a7-a27	📇 Networks
	Manuface Banhar
	2 Montol Probes
	Elasticity Actions
	Elesticity Actions
	Elasticity Actions User Applications
	Key Pairs Key Pairs Deployment Scripts
roperties 22 🤝 Authentication Token UI 🥅 Application Deployments	Key Pairs Geology Deployment Scripts
roperties 22 😽 Authentication Token UI 📇 Application Deployments	Key Pairs Key Pairs Deployment Scripts
roperties 😢 🤜 Authentication Token UI 📺 Application Deployments n Monitoring Probes	Key Pairs Key Pairs Deployment Scripts
roperties 😰 🤜 Authentication Token UI 📺 Application Deployments Monitoring Probes actly	Key Pairs Key Pairs Deployment Scripts
roperties 😢 🤜 Authentication Token UI 🦳 Application Deployments Monitoring Probes scity Acod Monitoring Probes	Hound Prodes Hound Prodes Elasticity Actions User Applications End Key Pairs Beployment Scripts
roperties 😰 👽 Authentication Token UI 🦳 Application Deployments n ficity VLoad litoring	Key Pairs Key Pairs Belloyment Scripts
roperties 12 C Authentication Token UI C Application Deployments Monitoring Probes bicity Wood itoring meters	Key Pairs Key Pairs Beployment Scripts
roperties 😰 🐢 Authentication Token UI 🦳 Application Deployments Monitoring Probes Authenticity Monitoring Probes Authenticity Pro	invenieur Product Elasticity Actions Elasticity Actions Elastic
roperties 😰 🛸 Authentication Token UI 🦳 Application Deployments Monitoring Probes itoring meters oyment	e Homor Prodes Elesticity Actions User Applications Beployment Scripts
roperties 😰 🛸 Authentication Token UI 🦳 Application Deployments Monitoring Probes Authentication Token UI 📄 Application Deployments Monitoring Probes Authentication Token UI 📄 Application Deployments Authentication Token UI 👘 Authentication Token UI 🁘 Authentication Deployments Authentication Token UI 👘 Application Deployments Authentication Token UI 👘 Authentication Deployments Authe	e Holmon Prodes E Elasticity Actions User Applications Beployment Scripts

Click on the Monitoring tab in the Properties view -> Create (Figure 24).



A java class that extends the JCatascopia Probe interface is automatically generated (Figure 25), guiding the user on how to implement the required methods. The created probes are exported as .jar files and are packaged in the CSAR file together with the application deployment file. At deployment time, the exported .jar files are installed on the specified VMs, just like the rest of the JCatascopia probes.

² http://linc.ucy.ac.cy/CELAR/jcatascopia/



```
Figure 25
```