The decision module actually contains the r-SYBL TUW tool for elasticity control.

1. R-SYBL System Configuration

1.1 Monitoring Plugin Configuration

1.1.1 Ganglia Plugin

For working with simple ganglia plugin some information has to be specified:

1.1.1.1 The ganglia plugin class needs to be specified in the config file (MonitoringPlugin = at.ac.tuwien.dsg.sybl.monitorandenforcement.monitoringPlugins.gangliaMonitoring.MonitoringGangliaAPI)

1.1.1.2 Ganglia IP needs to be specified in the config file

1.1.1.3 Ganglia Port needs to be specified in the config file

1.1.2 MELA Plugin

1.1.2.1 The main plugin class implementing MonitoringInterface needs to be specified in the config file (MonitoringPlugin = at.ac.tuwien.dsg.mela.apis.javaAPI.MELA_API)

1.1.2.2 ACCESS_MACHINE_IP needs to be specified (a machine having ganglia installed with the same port as the rest of the application)

1.1.2.3 GANGLIA_PORT needs to be specified

1.2 Enforcement Plugin Configuration

The **OpenStack plugin** class needs to be specified in the config file (EnforcementPlugin = at.ac.tuwien.dsg.sybl.monitorandenforcement.enforcementPlugins.openstack.EnforcementOpenstackAPI)

Some basic information need to be specified for being able to connect to the cloud for enforcing actions:

1.2.1 The name of the certificate used - CertificateName

1.2.2 The path towards the certificate - CertificatePath

1.2.3 Type of the cloud middleware used - CloudAPIType (in this case openstack-nova)

1.2.4 Endpoint for accessing the cloud infrastructure - CloudAPIEndpoint (in this case DSG local cloud http://openstack.infosys.tuwien.ac.at:5000/v2.0)

1.2.5 Username for accessing cloud infrastructure - CloudUser

1.2.6 Password for accessing cloud infrastructure – CloudPassword

2. Application Specific Configurations

The application-specific configuration can be sent to the rSYBL in three ways, depending on the information structure.

2.1 File-based configuration

2.1.1 The cloud service needs to be described, following the structure in the *serviceDescription.xml*(\analysis-engine\src\main\resources\config) example based on service units and service topologies.
SYBL supports elasticity requirements specified either through XML descriptions.

2.1.2 Elasticity requirements can be specified in an XML either separately (see *ElasticityRequirementsSpecification.xml* from analysisEngine\src\main\resources) or integrated in the service description file (as for the *serviceDescription.xml*). If specified in separate XML description file, this needs to be stated in the config file of the Control Service (SYBLDirectives = ./config/ElasticityRequirementsSpecification.xml).

2.1.3 The deployment description path (association between service units and snapshots, for being able to scale out when necessary) - DeploymentDescriptionPath (e.g. config/deploymentDescription.xml)

2.2 Runtime-based Configuration

2.2.1 Internal module based configuration

For the internal-module based configuration, the rSYBL service exposes a web method called `setApplicationDescriptionInfoInternalModel` which has the following parameters:

- the cloud service descriptor, XML description, received as String (see for example the content of the file serviceDescription.xml described in 2.1)
- the elasticity requirements which have not been added to the description of the cloud service, in XML form
- the deployment descriptor, XML description, received as String (see for example the content of the deploymentDescriptor.xml file discussed above)

2.2.2 TOSCA based configuration

For the internal-module based configuration, the rSYBL service exposes a web method called `setApplicationDescriptionInfoInternalModel` which has as parameter the TOSCA description, including the elasticity requirements injected in the TOSCA description at the necessary level (e.g. entire cloud service, topology/sub-topology level).

3. Development Guide

For customizing r-SYBL in what Monitoring and Enforcement is concerned, the following steps need to be followed.

For creating new plugins, an API of the plugin needs to implement the MonitoringInterface and respectively EnforcementInterface. The basic metrics and actions appear by name, while the ones which are specific to the applications/plugins are to be interfaced through getMetric and enforceAction actions.

Moreover, the needed configuration data needs to be added and processed from the config file of the Analysis Engine, which is the main module of rSYBL tool.

After adding new plugins, the rSYBL tool needs to be recompiled and re-deployed.