

# **SLA@SOI Open Reference Demonstrator Guide**

**Whitepaper**

**Version 1.0 July 28, 2011**

Contributors	
Partner	Contributors
XLAB	Primož Hadalin, Miha Stopar

Notices
<p>The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law. Copyright 2009 by the SLA@SOI consortium.</p>
<p>* Other names and brands may be claimed as the property of others.</p>



This work is licensed under a [Creative Commons Attribution 3.0 License](https://creativecommons.org/licenses/by/3.0/) [1].

Document History			
Version	Date	Author	Changes
0.1	24 May 2011	Primož Hadalin	Draft
0.2	23 June 2011	Miha Stopar	Minor changes, document skeleton is ready
0.3	19 July 2011	Miha Stopar	Some sections added (Studio...)
1.0	21 July 2011	Miha Stopar	Final corrections

# ***Executive Summary***

This document provides technical information on how the SLA@SOI Open Reference Demonstrator can be installed and started in order to demonstrate the SLA@SOI framework. It describes how to interact with the SLA@SOI framework through the Reference demonstrator UI and how to monitor various framework features.

Documents that are relevant to this guide are [SLA@SOI Adoption Guide](#) [2] which documents how to build and execute SLA@SOI framework for an arbitrary use case; [SLA@SOI Tutorial](#) [3] which documents how the SLA@SOI framework was applied for an Open Reference Case (ORC); [Reference Architecture guide](#) [4] which describes the architecture of the SLA@SOI framework; and [ORC deployment guide](#) [5].

# ***Table of Contents***

1	Introduction .....	6
2	Installation.....	7
2.1	Workload Generator .....	7
2.2	Open Reference Demonstrator GUI .....	7
3	How to start and use Open Reference Demonstrator Tools .....	9
3.1	SLA@SOI Framework Setup .....	9
3.2	Interaction with the SLA@SOI Framework via GUI .....	9
3.2.1	Login Screen .....	9
3.2.2	SLA Management.....	10
3.2.3	Framework Overview.....	13
3.3	Interaction with the Simulation Environment via Open Reference Demonstrator GUI .....	17
3.4	Customization and further development.....	21
4	References .....	22
	Appendix A: Glossary .....	23
	Appendix B: Abbreviations .....	26

# ***Table of Figures***

Figure 1:	Creating Release Build .....	8
Figure 2:	Login Screen .....	9
Figure 3:	User registration dialog .....	10
Figure 4:	Navigation Bar .....	10
Figure 5:	SLA Management.....	11
Figure 6:	SLA Hierarchy .....	11
Figure 7:	Create Agreement Step 1 (choosing a product) .....	12
Figure 8:	Create Agreement Step 2 (choosing a SLA template) .....	12
Figure 9:	Create Agreement Step 3 (renegotiation/create agreement) .....	13
Figure 10:	Framework Overview.....	14
Figure 11:	SLA Violations .....	15
Figure 12:	Infrastructure Overview .....	16
Figure 13:	Infrastructure node information popup window.....	17
Figure 14:	Selecting Workload Generator .....	18
Figure 15:	Configuring Workload Generator.....	18
Figure 16:	Response Times.....	19
Figure 17:	Arrival Rates .....	20
Figure 18:	Workload Generator Status .....	20

# **1 Introduction**

The purpose of the Reference demonstrator is to showcase the results of the scientific work performed on the Integrated European Research Project SLA@SOI.

The Reference demonstrator is based on Open Reference Case (ORC) - a reference application supporting the sales process in a retail-chain. ORC is implemented on top of legacy components and it was adapted to the service-oriented application. ORC was extended to support various deployment options and levels of separations as well as with additional software hooks that facilitated run-time monitoring and SLA-awareness of the application.

The Reference demonstrator is comprised of a graphical user interface that supports the interaction of the user with the SLA@SOI framework, of an ORC application, and of a workload generator that simulates the consumption of the ORC services. The GUI provides a view on a many aspects of the SLA@SOI framework, namely customer's perspectives (negotiation for the quality of ORC web services, re-negotiation, SLA warning, violations and penalties, status and state of the ORC services) and developer's perspectives (framework internal messages, SLA hierarchy, interaction between framework components).

As the Reference demonstrator is public and open source, the source code and documentation are publically available on [ORD Source Forge](#) [6].

This document provides detailed information about the steps needed to take in order to use Reference demonstrator.

## 2 Installation

### 2.1 Workload Generator

Pre-Requirements:

- Maven 3.0.\* [7]

The source code of the Workload Generator is located in the following SVN repository location:

<https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/ord/workload-generator/trunk>

The *manager.properties* file in the *workload* directory contains message bus properties and must be updated for the Workload Generator to connect to the PubSub server the framework uses.

Workload Generator can then be compiled and executed by running the following commands from the *workload* directory:

```
mvn compile
```

```
mvn exec:java -Dexec.mainClass="org.slasoi.adhoc.workload.WorkloadGenerator"
```

### 2.2 Open Reference Demonstrator GUI

Pre-Requirements:

- Maven 3.0.\* [7]
- Flex IDE (Flash Builder) [9]

The source code of the ORC web application is located in the following SVN repository location:

<https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/ord/gui/trunk>

The ORC Web Application consists of two projects: a server-based Java web application and a browser-based frontend implemented in [Adobe Flex](#) [8] which are located in *orc* and *orc-gui* folders, respectively.

First the ORC Java web application must be deployed to a Java Servlet container such as [Apache Tomcat](#) [10]. The application's pom.xml file already includes a plugin for deployment, though the URL must be updated to point to the server:

```
<groupId>org.codehaus.mojo</groupId>
<artifactId>tomcat-maven-plugin</artifactId>
<configuration>
    <url>http://172.16.117.118:8080/manager</url>
```

Note: if a Tomcat server is used then Maven's settings file (settings.xml), which is usually located in .m2 directory, must contain the user name and password of the user with manager privileges so that the deployment plugin is authorized:

```
<server>
    <id>tomcat</id>
    <username>manager-username</username>
    <password>manager-password</password>
</server>
```

To deploy the application run the following Maven command:

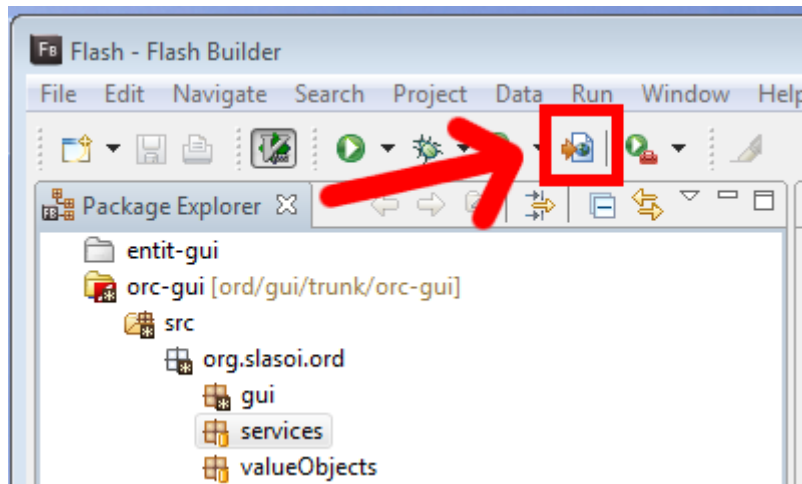
```
mvn clean compile war:war tomcat:deploy
```

After the deployment to a Java Servlet server the `orc-gui` project must be imported into Flex development environment and project server properties must be updated (in Flash Builder go to Project > Properties > Flex Server).

It might be necessary to update endpoint addresses the ORC application uses to communicate with the framework. Change the addresses in the properties file located on:

```
[Servlet Server's directory containing web applications]/orc/WEB-INF/classes/config/framework.properties
```

The last step is to create a release build of the project. To do this in Flash Builder go to Project > Export Release Build (Figure 1).



**Figure 1: Creating Release Build**

Apache Tomcat contains a directory named `webapps` which resides in Tomcat's installation directory by default. Once the project is uploaded to `webapps` directory it must be started by restarting the server or using Tomcat's manager web interface.

# 3 How to start and use Open Reference Demonstrator Tools

## 3.1 SLA@SOI Framework Setup

SLA@SOI framework setup is documented in [SLA@SOI Adoption Guide](#) [2] which documents how to build and execute SLA@SOI framework for an arbitrary use case.

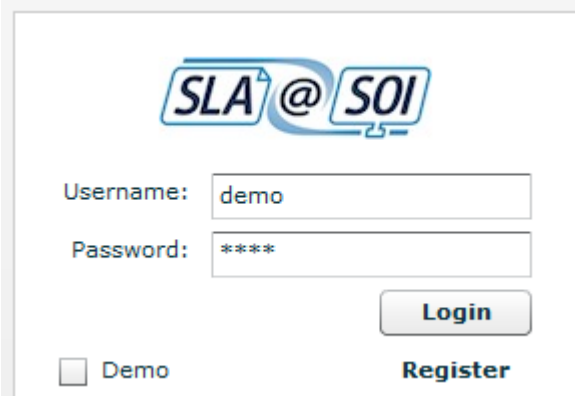
A detailed documentation on how SLA@SOI framework was applied for an Open Reference Case (ORC) is available within [SLA@SOI Tutorial](#) [3].

## 3.2 Interaction with the SLA@SOI Framework via GUI

### 3.2.1 Login Screen

Login screen (Figure 2) is the first screen the user interacts with. Registered users authenticate themselves against the user database by entering their username and password, while new users register by clicking the *Register* button which opens a registration dialog (Figure 3). The GUI communicates with Business Manager which is responsible for managing user data.

The *Demo* checkbox enables demo mode meaning the demonstrator doesn't actually communicate with the framework but rather uses static mock data. This is used for demonstrational purposes when the framework is not available due to e.g. network connectivity issues.



The screenshot shows a login interface for SLA@SOI. At the top center is the logo, which consists of the text 'SLA@SOI' in a stylized font with a blue and white color scheme. Below the logo, there are two text input fields. The first is labeled 'Username:' and contains the text 'demo'. The second is labeled 'Password:' and contains the text '\*\*\*\*'. To the right of the password field is a button labeled 'Login'. At the bottom left of the form area, there is a checkbox labeled 'Demo'. At the bottom right, there is a button labeled 'Register'.

Figure 2: Login Screen

The 'Register' dialog box contains the following fields and controls:

- First Name:
- Last Name:
- Address:
- Country:
- Phone:
- Fax:
- Email:
- Job Department:
- Job Title:
- Currency:
- Language:
- Username:
- Password:
- Confirm Password:

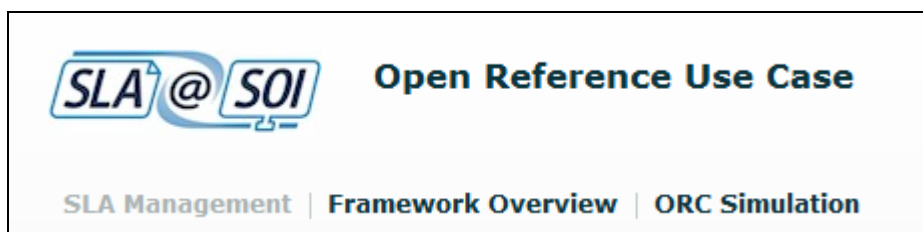
A 'Register' button is located at the bottom center of the dialog.

**Figure 3: User registration dialog**

### **3.2.2 SLA Management**

After successful authentication/registration the GUI establishes a connection to the framework event bus and performs other initialization steps.

The GUI has a static header and a navigation bar (Figure 4) at the top to switch between different sections of the GUI. SLA Management (Figure 5) is the section the user lands on after the authentication. The other views of the GUI are Framework Overview and ORC Simulation.



**Figure 4: Navigation Bar**

SLA Management

Name	Template ID	SLA Hierarchy	Agreed At	Effective From	Effective Until	Status
ORC test	ORC Template	View	Fri, 27.05.2011 13:33	Fri, 27.05.2011 13:33	Wed, 01.06.2011 13:33	expired
ORC SLA	ORC Template	View	Thu, 09.06.2011 1:33	Thu, 09.06.2011 1:33	Sun, 26.06.2011 15:33	agreed

Create Agreement

**Figure 5: SLA Management**

SLA Management section shows a list of customer's SLAs along with some SLA properties: Name, Template ID, SLA Hierarchy, Agreed At, Effective From, Effective Until and Status.

SLA Hierarchy column contains a *View* button which shows all SLAs associated to the agreement: Business SLA, Software SLA and Infrastructure SLA. Figure 6 shows contents of Business SLA of the particular agreement.

Business SLA

SLA Hierarchy
View
View

```

role = customer
}
/* ----- INTERFACE DECLARATIONS----- */
interface_declr{
  id = ORCPaymentService
  provider_ref = ORCProvider
  interface_spec{
    name = PaymentService
    operation{
      id = PaymentServiceOperation
      input{
        name = cardInformation
        datatype = xsd:string
        auxiliary = false
      }
      input{
        name = cardNumber
        datatype = xsd:string
        auxiliary = false
      }
      output{
        name = PaymentServiceResponse
        datatype = xsd:string
        auxiliary = false
      }
    }
  }
}
}
}

```

**Figure 6: SLA Hierarchy**

The user can create an agreement (new SLA) by clicking the *Create Agreement* button. This opens a wizard which guides the user through the SLA creation process.

SLA creation process consists of several steps. The first step is to choose a product from the list (Figure 7).

**Add New SLA Agreement** [X]

Choose a product

**Products**

Name	Brand	Category
ORC		inventory, paymen

**Product Details**

**Product Description**

Open Reference Case

**Product Offers**

Name	Description	Services
Inventory	Inventory	Get Product Details, Book Sal
Payment	Payment	Paymen

Next Cancel

**Figure 7: Create Agreement Step 1 (choosing a product)**

The second step is to choose a SLA template from the list (Figure 8). When a template is chosen a list of agreement terms is shown and the user has an option to change the values by clicking the *Value* table cell in the *Guaranteed States* table. The table cell becomes editable thus allowing the user to change the value.

To see the textual representation of a SLA template, the user must click the *View* button in the SLA templates table.

**Add New SLA Agreement** [X]

Choose SLA Template

**SLA Templates**

ID	Model Version	
ORC_BusinessSLAT	sla_at_soi_sla_model_v1.0	View

**SLA Template Details**

**Agreement Terms**

- ORC\_CustomerConstraintPayment
- ORC\_CustomerConstraintInventoryGetDet
- ORC\_CustomerConstraintInventoryBookSi
- ORC\_ResponseTimeInventoryBookSale
- ORC\_ResponseTimeInventoryProductDetail
- ORC\_ResponseTimePayment
- ORC\_ReliabilityPayment

**Guaranteed States**

ID	State	Value
ORCThroughputConstraintPayment	arrival_rate(ORCPaymentService/PaymentService	70

**Guaranteed Actions**

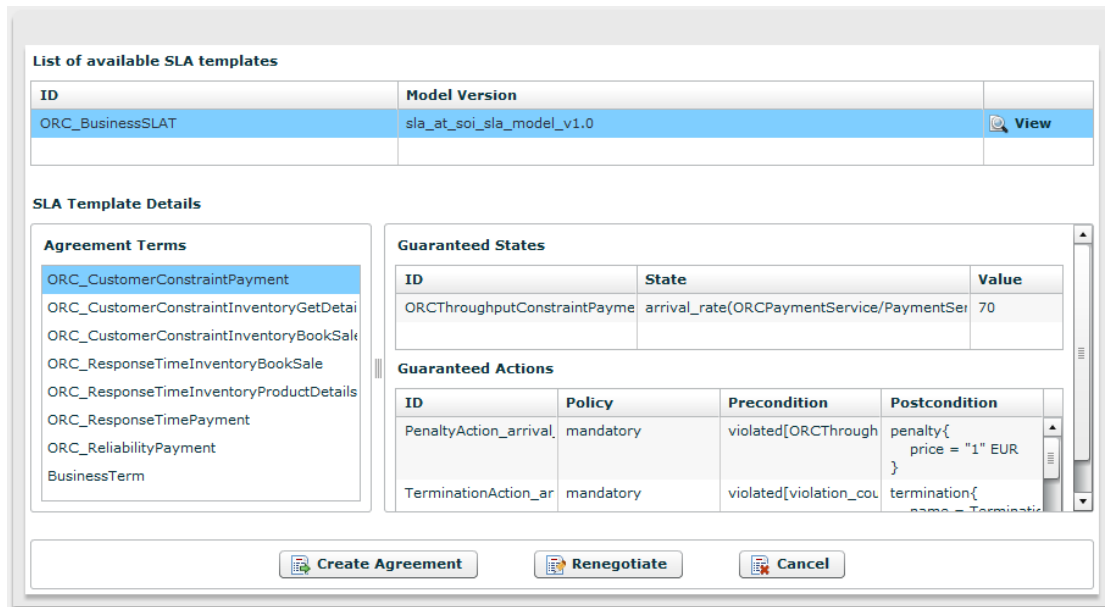
ID	Policy	Precondition	Postcondition
PenaltyAction_arrival_r	mandatory	violated[ORCThroughpu	penalty{ price = "1" EUR }
TerminationAction_arriv	mandatory	violated[violation_count	termination{ name = Termination terminationClauseId

Previous Finish Cancel

**Figure 8: Create Agreement Step 2 (choosing a SLA template)**

Clicking the *Finish* button starts the negotiation process. The user can also go back to previous step to choose another product by clicking the *Previous* button or cancel agreement creation altogether by clicking the *Cancel* button.

After the framework finishes the negotiation, it returns a list of agreeable SLA templates. The user has an option to either change the values of the SLA template again and renegotiate or create an agreement based on the selected SLA template (Figure 9).



**Figure 9: Create Agreement Step 3 (renegotiation/create agreement)**

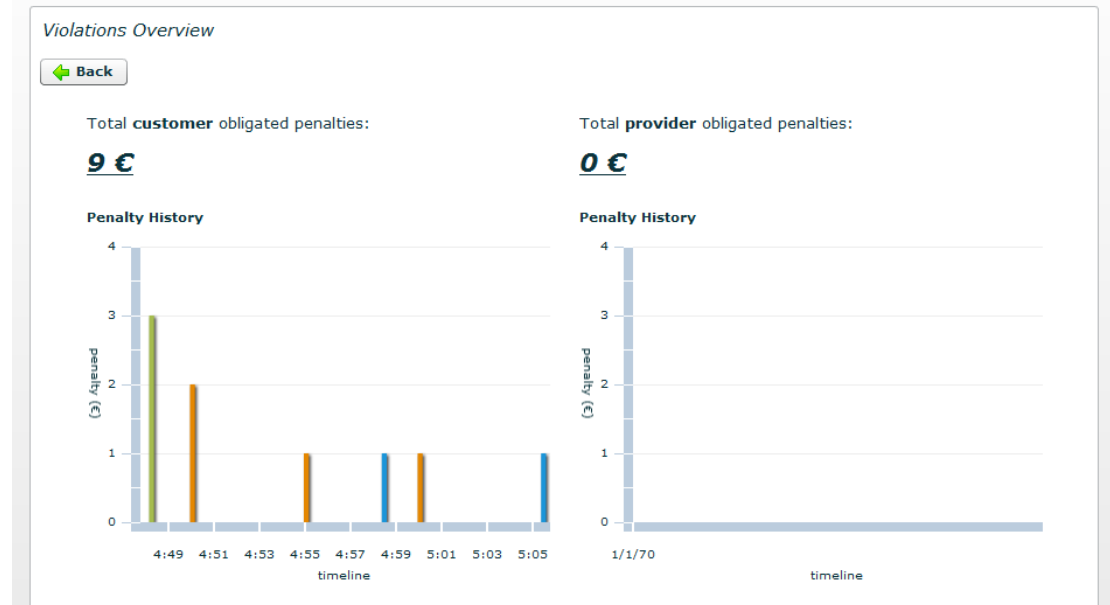
In the case of renegotiation the same window appears again after the user has clicked the *Renegotiate* button. This can be done several times until the customer is satisfied with the offer.

Clicking *Create Agreement* button is the last step of the agreement creation process. The framework creates an agreement (SLA) based on the selected SLA template, stores the SLA into the SLA registry and returns the SLA to the GUI which is added to the table in SLA Management section.

### 3.2.3 Framework Overview

The purpose of the Framework Overview section is to give feedback to the user about the framework in runtime (Figure 10).

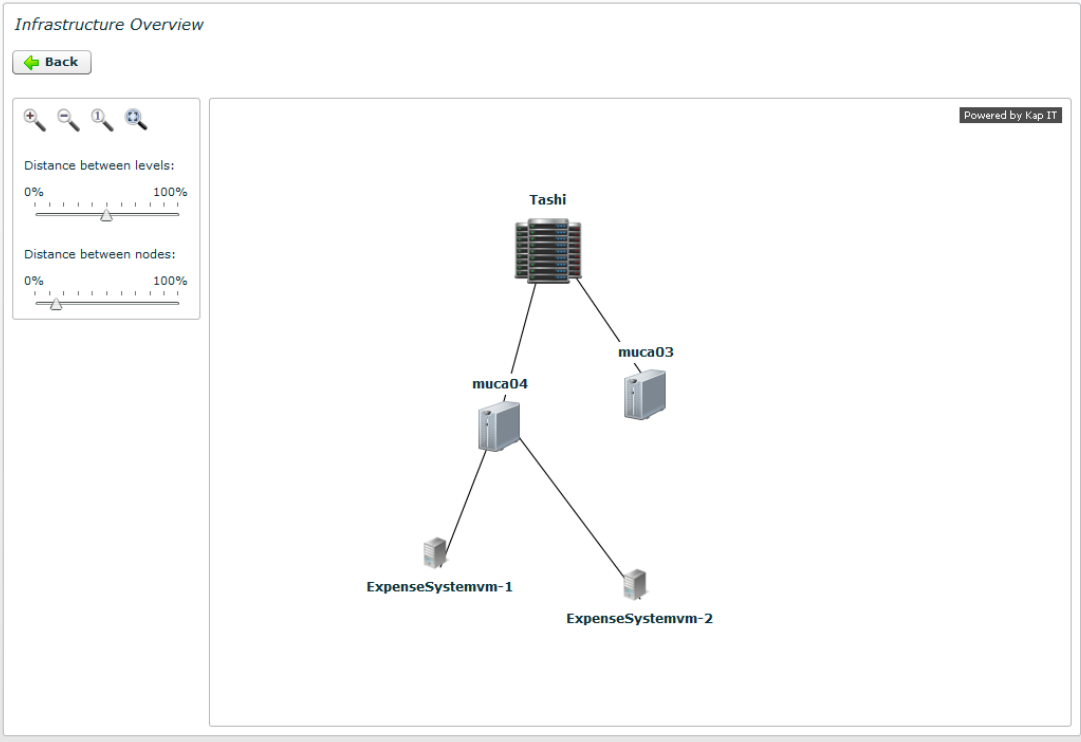




**Figure 11: SLA Violations**

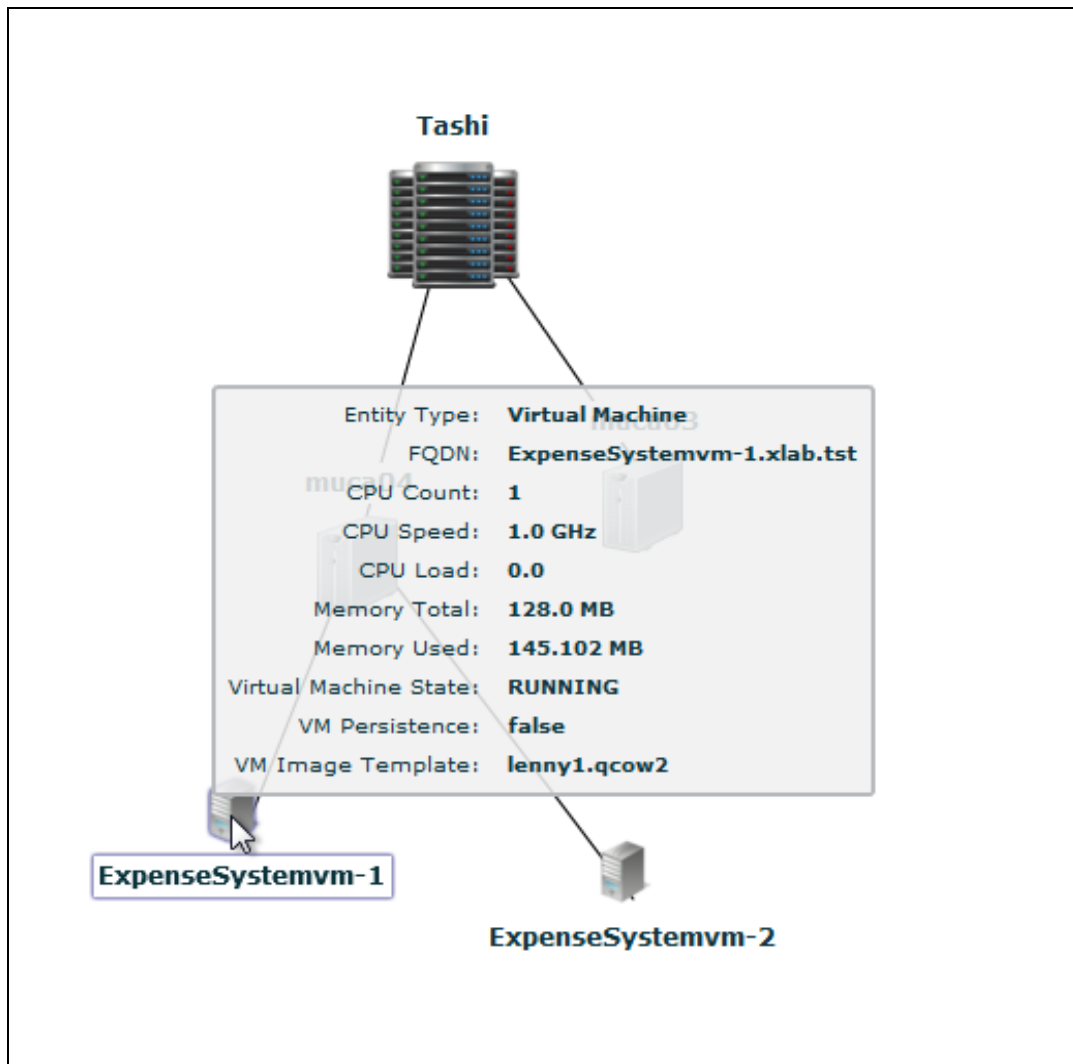
After service provisioning the monitoring of the service is established and any SLA violation is detected and reported. Violations Overview displays customer and provider obligated penalties along with violations history graphs.

Clicking the *Infrastructure Service Manager* component opens Infrastructure Overview section that gives detailed information about the infrastructure (Figure 12).



**Figure 12: Infrastructure Overview**

Controls on the left side of the Infrastructure Overview are used to control the infrastructure graph: zooming, distance between the nodes, etc. By moving mouse pointer over any node in the infrastructure graph, a popup window containing detailed information of the node is shown (Figure 13). The nodes are organized in a hierarchical manner from cluster (top) to hosts (middle) and further down to virtual machines (bottom). Although the infrastructure graph gets updated automatically, it is possible to manually refresh the graph by right-clicking on infrastructure graph and selecting *Synchronize* from the context menu.



**Figure 13: Infrastructure node information popup window**

### **3.3 Interaction with the Simulation Environment via Open Reference Demonstrator GUI**

ORC Simulation provides means to simulate a running Open Reference Case (ORC) application. This is done using a Java application named Workload generator that interacts with the provisioned services.

Upon opening ORC Simulation view a user must select a workload generator from the list of available Workload generators (Figure 14). If no Workload generators are running, one can be started by clicking the *Remote Start* button.

Workload Generator Selection

Online Workload Generators

1308651014292

Select Remote Start

**Figure 14: Selecting Workload Generator**

After Workload generator has been selected a screen for configuring is opened (Figure 15).

Workload Management | Response Times | Arrival Rates | Status

Workload Generator Configuration & Management

ORC Endpoint URI:

**Simulation Profile**

Number of Cash Desks:  Number of Bought Products:

Random Seed:

**Simulation Delays and Deviations**

Auto Adjust Delay: Short  Long

Wait For Customer:   Scan:

Capture Manually:   Update Running Total:

Give Card:   Manual Payment:

Wait to Process:

**Simulation Probabilities**

Data Recognized:  Payment With Credit Card:

Card Valid:

**Simulation Management**

Configure

Start Stop Pause Resume

Change Workload Generator

**Figure 15: Configuring Workload Generator**

The first parameter is ORC Endpoint URI which gets automatically set upon agreement creation. It is an address that the Workload generator uses to invoke ORC methods through web services interface.

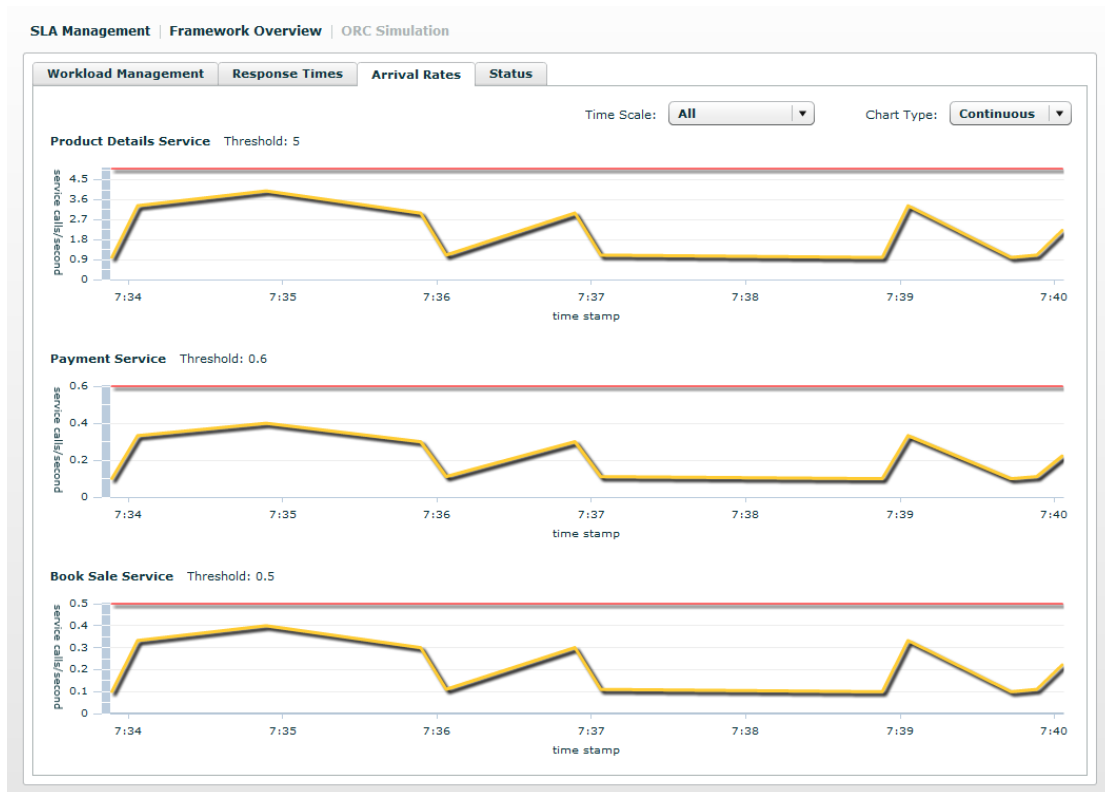
There are many parameters available to be set that affect how Workload generator interacts with the provisioned services. The ones that affect the arrival rate are under *Simulation Delays and Deviations* section. Instead of manually setting each parameter, *Auto Adjust Delay* slider can be used to automatically set simulation delay.

After the parameters have been set, the workload is configured by clicking the *Configure* button. The functionality behind Simulation Management buttons (*Configure*, *Start*, *Stop*, *Pause*, *Resume*) prevent the user from misusing the Workload Generator by automatically enabling and disabling the buttons. The simulation is started by pressing the *Start* button.

Once the simulation is running the framework's monitoring service will begin reporting response times and arrival rates which are represented in *Response Times* (Figure 16) and *Arrival Rates* (Figure 17) graphs respectively.



**Figure 16: Response Times**



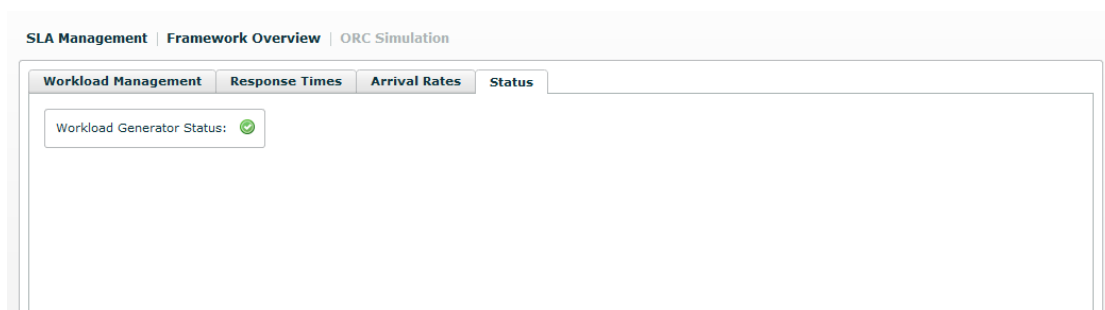
**Figure 17: Arrival Rates**

There are three graphs in each view, one for every service: Product Details, Payment and Book Sale service.

Both *Response Times* and *Arrival Rates* views share similar functionality. At the top there are two controls that affect how the data is represented in the graphs. *Time Scale* control is used to define a time scale of the data represented in the graphs. There are three options: *All*, *Last 5 minutes* and *Last minute*. *Chart Type* control is used to control whether the data is represented in a continuous (line chart) or discrete (column) manner.

The agreed-upon SLA contains agreement terms that define their violation threshold values. These thresholds are represented in graphs as horizontal red line. Whenever a threshold line has been crossed, a violation occurred and logged in *Violations Overview* section.

The last tab in ORC Simulation navigation bar is titled *Status* and is used to indicate whether Workload generator is alive or not (Figure 18).



**Figure 18: Workload Generator Status**

## **3.4 Customization and further development**

In case that somebody would like to customize the framework and test it from the Reference demonstrator UI, the documentation about the SLA@SOI Studio [11] might be helpful. SLA@SOI Studio is an Eclipse based plugin, developed to simplify the SLA@SOI framework adoption for an arbitrary use case. It enables simple importing of the common SLA@SOI packages, their customization through the step-by-step guides, it enables SLA@SOI framework debugging, its configuration and execution. The Studio source code, as well as the [documentation](#) [11], can be found on [Source Forge](#) [12].

## 4 References

- [1] Creative Commons Attribution 3.0 License. URL: <http://creativecommons.org/licenses/by/3.0/>
- [2] Adoption guide on SourceForge. URL: [https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/platform/trunk/doc/SLA@SOI-Adoption\\_Guide.doc](https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/platform/trunk/doc/SLA@SOI-Adoption_Guide.doc)
- [3] Tutorial on SourceForge. URL: <https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/platform/trunk/doc/SLA@SOI-tutorial.doc>
- [4] Reference Architecture on SourceForge. URL: [http://sourceforge.net/apps/trac/sla-at-soi/browser/platform/trunk/doc/SLA%40SOI-Reference\\_Architecture.pdf](http://sourceforge.net/apps/trac/sla-at-soi/browser/platform/trunk/doc/SLA%40SOI-Reference_Architecture.pdf)
- [5] ORC deployment guide on SourceForge. URL: [https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/ord/doc/ORC\\_Deployment\\_Guide.pdf](https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/ord/doc/ORC_Deployment_Guide.pdf)
- [6] Open Reference Demonstrator source code on SourceForge. URL: <http://sla-at-soi.svn.sourceforge.net/viewvc/sla-at-soi/ord/>
- [7] Apache Maven. URL: <http://maven.apache.org/>
- [8] Adobe Flex. URL: <http://www.adobe.com/products/flex/>
- [9] Adobe Flash Builder. URL: <http://www.adobe.com/products/flash-builder.html>
- [10] Apache Tomcat. URL: <http://tomcat.apache.org/>
- [11] Studio use guide on SourceForge. URL: [https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/platform/trunk/doc/Studio-User\\_Guide.doc](https://sla-at-soi.svn.sourceforge.net/svnroot/sla-at-soi/platform/trunk/doc/Studio-User_Guide.doc)
- [12] Studio source code. URL: <http://sourceforge.net/apps/trac/sla-at-soi/browser/studio/trunk>

# Appendix A: Glossary

The following list shows the most important entries of the SLA@SOI glossary. Note that terms that are specific for the current document and not part of the overall project wide glossary are marked with an asterix \*.

Agreement Initiator	An agreement initiator is a party to a <i>service level agreement</i> . The initiator creates and manages an agreement on the availability of a service on behalf of either the service customer or service provider, depending on the domain-specific signalling requirements.
Agreement Offer	An offer is the description of the agreement relationship that is sent from <i>agreement initiator</i> to <i>agreement responder</i> during agreement creation, indicating the relationship which the initiator would like to form.
Agreement Responder	The agreement responder is a party to a <i>service level agreement</i> . The responder implements and exposes an agreement on behalf of either the service provider or service customer, depending on the domain-specific signalling requirements.
Agreement Template	An agreement template is an XML document used by the <i>agreement responder</i> to advertise the types of offers it is willing to accept.
Agreement Term	Agreement terms define the content of a <i>service level agreement</i> .
Business Service	A business service is exposed/invoked via at least some non IT elements.
Business Manager	A specialization of <i>service provider</i> : person that defines the SLATs of products and joins available services in a product.
External Service	External services are exposed across the boundaries of an organization, i.e. across at least two administrative domains.
Framework Administrator	A specialization of <i>service provider</i> : person that configures/adapts the SLA@SOI framework for a specific application.
Guarantee Term	Guarantee terms define the assurance on service quality associated with the service described by the service definition terms. They refer to the service description that is the subject of the agreement and define service level objectives, qualifying conditions and business value expressing the importance of the service level objectives.
Hybrid Service	A hybrid service is a set or bundle of other services where all these services are exposed to the customer

	but have different service interface types (e.g. an IT service and a business service).
Infrastructure Manager	A specialization of <i>infrastructure provider</i> : person/system that is interested to measure and control infrastructure properties.
Infrastructure Provider	A specific kind of service provider that focuses on the provisioning of <i>infrastructure services</i> .
Infrastructure Service	An infrastructure service is a specific <i>IT service</i> which exposes resource/hardware-centric capabilities.
Internal Service	Internal services are exposed within the boundaries of an organization, i.e. within one administrative domain.
IT Service	An IT service is exposed/invoked by means of information technology. Specific classes of IT services may be software services, infrastructure services or media services.
Offered Service	An abstract service (more precisely: service type) which is offered by a specific <i>Service Provider</i> to its <i>Service Customers</i> .
Operation Level Agreements	A specification of the conditions under which an <i>internal service</i> or a component is to be used by its "customer".
Service	A means of delivering value to customers by facilitating outcomes customers want to achieve without the ownership of specific costs and risks. See also <i>service interface type</i> , <i>service concreteness</i> , <i>service exposure</i>
Service Concreteness	The stage a service reaches over time from a fully abstract type to actually instantiated. See also <i>service type</i> , <i>offered service</i> , <i>service implementation</i> , <i>service instance</i>
Service Consumer	Person(s) who actually consume/use the provided services. Typically they belong to the <i>service customer</i> .
Service Customer	Someone (person or group) who orders/buys services and defines and agrees the service level targets.
Service Description Term	Service Description Terms describe the functionality that will be delivered under the <i>service level agreement</i> . The agreement description may include also other non-functional items referring to the service description terms.
Service Exposure	Services can be exposed either internally (within the same administrative domain) or externally. See also <i>internal service</i> , <i>external service</i>
Service Implementation	A service implementation is a possible concrete realization of a given <i>service type</i> .
Service Instance	A concrete realization of an <i>offered service</i> which is ready for consumption by service users. It relies on the instantiations of all the resources required for a given <i>service implementation</i> .

Service Interface Type	Describes the nature of an actually exposed service, i.e. about the nature of his invocation interface. See also <i>business service</i> , <i>IT service</i> , <i>hybrid service</i>
Service Level Consequence	An action that takes place in the event that a service level objective is not met.
Service Level Agreement	An agreement defines a dynamically-established and dynamically managed relationship between parties. The object of this relationship is the delivery of a service by one of the parties within the context of the agreement. The management of this delivery is achieved by agreeing on the respective roles, rights and obligations of the parties. The agreement may specify not only functional properties for identification or creation of the service, but also non-functional properties of the service such as performance or availability. Entities can dynamically establish and manage agreements via Web service interfaces.
Service Level Objective	Service Level Objective represents the quality of service aspect of the <i>agreement</i> . Syntactically, it is an assertion over the agreement <i>terms</i> of the agreement as well as such qualities as date and time.
Service Provider	An organization supplying services to one or more internal customers or external customers.
SLA Manager	A specialization of <i>service provider</i> : person/system that is responsible for managing SLATs and SLA relationships.
Software Designer	A specialization of <i>software provider</i> : person that designs/develops the architecture and components of a specific SLA based application.
Software Manager	A specialization of <i>service provider</i> : person that defines software-based services, takes care of their management and supports the SLA manager in creating appropriate SLA templates.
Software Provider	An organization producing <i>software components</i> which might be used by a <i>service provider</i> to assemble actual <i>services</i> .
Software Service	A software service is a specific <i>IT service</i> which is exposed/invoked by means of software entities such as Web services, user interfaces, or software-based business processes.
Software Component	Software components are the entities produced at design-time by a <i>software provider</i> .
Service Type	A service type (or abstract service) specifies the external interface of a service possibly including non-functional aspects. It does not specify any means (components, resources) which are needed for the actual provisioning of that service.

## ***Appendix B: Abbreviations***

AOP	Aspect Oriented Programming
BM	Business Manager
B-SLAM	Business SLA Manager
EMF	Eclipse Modelling Framework
ERP	Enterprise Resource Planning
IE	Interaction Event
FCR	Finite capacity regions
ISLAM	Infrastructure SLA Manager
ISM	Infrastructure Service Manager
IoC	Inversion of Control
KPI	Key Performance Indicator
LLMS	Low Level Monitoring System
LQN	Layered Queueing Networks
MA	Manageability Agent
MRE	Monitoring Result Event
MVC	Model View Controller
NFP	Non-functional property
ORC	Open Reference Case
OVF	Open Virtualization Format
QoS	Quality of Service
QPN	Queueing Petri Nets
PAC	Provisioning and Adjustment Component
POC	Planning and Optimization Component
POJO	Plain Old Java Objects
SaaS	Software as a Service
SE	Service Evaluation
SLA	Service Level Agreement
SLAM	SLA Manager
SLAT	Service Level Agreement Template
SM	Service Manager
SME	Small and Medium-sized Enterprise
SOA	Service Oriented Architecture
SW-SLAM	Software SLA Manager
SW-SM	Software Service Manager
TCO	Total Cost of Ownership
TOGAF	The Open Group Architecture Framework