Project Acronym:	**OPTIMIS**
Project Title:	**Optimized Infrastructure Services**
Project Number:	**257115**
Instrument:	**Integrated Project**
Thematic Priority:	**ICT-2009.1.2 – Internet of Services, Software and Virtualisation**

Provider's Risk Assessment Tools Installation Guide

Activity 3: *Service Deployment*

WP 3.4: *Provider's Risk Assessment Tools*

Due Date:		M34
Submission Date:		15/03/2013
Start Date of Project:		01/06/2010
Duration of Project:		36 months
Organisation Responsible for the Deliverable:		University of Leeds
Version:		1.0
Status		Final
Author(s):	Ming Jiang	ULEEDS
	Thomas Kirkham	ULEEDS
	Karim Djemame	ULEEDS

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission)	
RE	Restricted to a group specified by the consortium (including the Commission)	
CO	Confidential, only for members of the consortium (including the Commission)	

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	08/03/2013	First draft	Ming Jiang
1.0	12/03/2013	Final Version	Ming Jiang

Table of Contents

Index of Tables

1 Introduction

This document includes the installation guide for the software component Provider's Risk Assessment Tools: Service Provider Risk Assessment Tool (SPRAT) and Infrastructure Provider Risk Assessment Tool (IPRA).

The SPRAT is responsible for supporting the risk-aware negotiation with IPs on behalf of end-users. The risk assessments of SPRAT are conducted in 3 of the total 6 stages of SPRAT and IPRAT:

- SPRAT's Stage 1 will involve the Service Deployment Optimizer (SDO) invoking the SPRAT to collect some basic data about the IP's available for use. After receiving this data from the monitoring tool, the SDO will then be able to request a risk factor of working with each of the IPs available.
- SPRAT's Stage 4 will involve the IP telling the SP about the risk it estimates of failure if the service is deployed on it. The SDO on the SP side will read in this detail and calculate an Adjusted Probability of Failure (APoF) using its own historical data. The SDO will then make a decision on where to deploy the service.
- SPRAT's Stage 5 involves dynamic assessment of the service on the SP level.

The IPRAT estimates risk for an Service Level Agreement (SLA) violation and supports the IP's decision of agreeing an SLA as well as of initiating fault-tolerance mechanisms to prevent SLA violation. Risk assessment conducted by IPRAT improves the IP's reliability and QoS. The risk assessments of IPRAT are conducted in 3 of the total 6 stages of SPRAT and IPRAT:

- IPRAT' stages 2 and 3 together are able to collect some basic information about the SP in question. Using this information the IPRA can send results back to the Admission Controller (AC) to give an indication of the risk of working with that SP with the Probability of Failure (PoF) of the Service Manifest is to be deployed.
- IPRAT's stage 6 is able to proactively conduct the dynamic assessment of the service during operation on the IP. During this phase the monitoring tool will be constantly observing the service collecting various data such as the resource usage. This data will be read in by the IPRAT and returns the mitigation strategies (such as Cloud Bursting) to the CO/HM.

1.1 Glossary of Acronyms

Acronym	Definition
AC	Admission Controller
APoF	Adjusted Probability of Failure
CO	Cloud Optimizer
HM	Holistic Manager
IPRAT	Infrastructure Provider Risk Assessment Tool
PoF	Probability of Failure
SDO	Service Deployment Optimizer
SPRAT	Service Provider Risk Assessment Tool
SLA	Service Level Agreement

2 SPRAT Installation Guide

2.1 Release information

Component Name	Release Number	Release Date
Service Provider Risk Assessment Tool (SPRAT)	1.0-SNAPSHOT	15/03/2013

2.2 Minimal System Requirements

No special requirements.

2.3 Platforms Supported

The component supports any Java-compliant platform.

2.4 Software Pre-requisites and Dependencies

Table 1 SPRAT Software Dependencies

```
<dependency>
    <groupId>commons-httpclient</groupId>
    <artifactId>commons-httpclient</artifactId>
    <version>3.1</version>
</dependency>
<dependency>
    <groupId>commons-logging</groupId>
    <artifactId>commons-logging</artifactId>
    <version>1.1.1</version>
</dependency>
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.14</version>
</dependency>
<dependency>
    <groupId>com.sun.jersey</groupId>
    <artifactId>jersey-client</artifactId>
```

```
        <version>1.6</version>
      </dependency>
      <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>3.8.1</version>
        <scope>test</scope>
      </dependency>
<!--put log4j at the top of the dependency list-->
      <dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version>1.2.14</version>
      </dependency>
      <dependency>
        <groupId>eu.optimis.trec.common.db</groupId>
        <artifactId>TRECCommonDBIP</artifactId>
        <version>3.0-SNAPSHOT</version>
      </dependency>

     <!-- Monitoring resource -->
      <dependency>
        <groupId>eu.optimis.mi.monitoringresource</groupId>
        <artifactId>MonitoringResource</artifactId>
        <version>0.0.1-SNAPSHOT</version>
      </dependency>

      <!-- REST client for Monitoring -->
      <dependency>
        <groupId>eu.optimis.mi.restclient</groupId>
        <artifactId>RESTClient</artifactId>
        <version>0.0.2-SNAPSHOT</version>
      </dependency>
```

```
<!-- REST client for CO/HM -->
<dependency>
  <groupId>eu.optimis</groupId>
  <artifactId>CloudOptimizerRESTClient</artifactId>
  <version>1.0-SNAPSHOT</version>
</dependency>
<!-- REST client for FTE -->
<dependency>
  <groupId>eu.optimis</groupId>
  <artifactId>FaultToleranceEngineRESTClient</artifactId>
  <version>1.0-SNAPSHOT</version>
</dependency>
<!-- Hibernate framework -->
<dependency>
  <groupId>org.codehaus.mojo</groupId>
  <artifactId>hibernate3-maven-plugin</artifactId>
  <version>2.2</version>
</dependency>

<!-- Hibernate library dependecy start -->
<dependency>
  <groupId>dom4j</groupId>
  <artifactId>dom4j</artifactId>
  <version>1.6.1</version>
</dependency>
<dependency>
  <groupId>commons-logging</groupId>
  <artifactId>commons-logging</artifactId>
  <version>1.1.1</version>
</dependency>
<dependency>
  <groupId>commons-collections</groupId>
  <artifactId>commons-collections</artifactId>
  <version>3.2.1</version>
</dependency>
```

```
<dependency>
  <groupId>commons-configuration</groupId>
  <artifactId>commons-configuration</artifactId>
  <version>1.5</version>
</dependency>
<dependency>
  <groupId>cglib</groupId>
  <artifactId>cglib</artifactId>
  <version>2.2</version>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>1.5.2</version>
  <type>jar</type>
  <scope>compile</scope>
</dependency>
<!-- Hibernate library dependecy end -->

<dependency>
  <groupId>javax.persistence</groupId>
  <artifactId>persistence-api</artifactId>
  <version>1.0</version>
</dependency>
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>3.8.1</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.6</version>
</dependency>
```

```
<dependency>
  <groupId>hsqldb</groupId>
  <artifactId>hsqldb</artifactId>
  <version>1.8.0.7</version>
</dependency>
<dependency>
  <groupId>eu.optimis.servicemanifest</groupId>
  <artifactId>service-manifest-api</artifactId>
  <version>1.0.8</version>
</dependency>
<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-math</artifactId>
  <version>2.0</version>
</dependency>
<dependency>
  <groupId>com.sun.jersey</groupId>
  <artifactId>jersey-server</artifactId>
  <version>1.6</version>
</dependency>
<dependency>
  <groupId>com.sun.jersey</groupId>
  <artifactId>jersey-core</artifactId>
  <version>1.6</version>
</dependency>
<dependency>
  <groupId>com.sun.jersey</groupId>
  <artifactId>jersey-client</artifactId>
  <version>1.6</version>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
```

```
    </dependency>
</dependencies>
```

The dependencies listed in Table 1 are included the pom files of the module maintained by the Apache Maven 3.0.3 automatically.

2.5 Installation Instructions

- SVN path: http://pandora.atosorigin.es/svn/optimis/branches/OptimisY3/ServiceProviderRiskAssessmentTool
- The SPRAT Client API competent (as a jar file) is maintained by the maven development environment; the following dependency should be included into the maven pom.xml file.

```
<dependency>
        <groupId>eu.optimis</groupId>
        <artifactId>ServiceProviderRiskAssessmentToolRESTClient</artifactId>
        <version>1.0-SNAPSHOT</version>
</dependency>
```

Please note SPRAT API is part of the TRECCommonAIPSP and can be used with TRECCommonAPISP together:

```
<dependency>
        <groupId>eu.optimis.trec.common</groupId>
        <artifactId>TRECCommonAPISP</artifactId>
        <version>3.0-SNAPSHOT </version>
 </dependency>
```

- The SPRAT Web Service competent is maintained by the maven development environment; the following dependency should be included into the maven pom.xml file.

```
<dependency>
        <groupId>eu.optimis</groupId>
        <artifactId> ServiceProviderRiskAssessmentToolRESTService </artifactId>
        <version>1.0-SNAPSHOT</version>
 </dependency>
```

The SPRAT Web Service competent can be built into a WAR file and deployed into the Tomcat web application container (e.g. /usr/local/apache-tomcat-6.0.32/webapps/).

2.6 Getting started

2.6.1 Using the Software

Once the SPRAT Web Service is deployed into the Tomcat, the SPRAT Client API will connect to it based on the configuration information defined by the optimis.properties file in /opt/optimis/etc.

2.6.2 Testing the Software

Java Unit Test cases are provided and can be executed in the maven environment by run the 'mvn test' command in the ServiceProviderRiskAssessmentTool directory.

2.6.3 Configuration

- log4jSPRA.properties in /opt/optimis/etc/ServiceProviderRiskAssessmentTool
- Log file: /opt/optimis/var/log/ServiceProviderRiskAssessmentTool/ServiceProviderRiskAssessmentTool.log

2.7 FAQ

3 IPRAT Installation Guide

3.1 Release information

Component Name	Release Number	Release Date
Infrastructure Provider Risk Assessment Tool (IPRAT)	1.0-SNAPSHOT	15/03/2013

3.2 Minimal System Requirements

No special requirements.

3.3 Platforms Supported

The component supports any Java-compliant platform.

3.4 Software Pre-requisites and Dependencies

Table 2 IPRAT Software Dependencies

```
<dependencies>
    <!-- Monitoring resource -->
    <dependency>
      <groupId>eu.optimis.mi</groupId>
      <artifactId>MonitoringInfrastructure</artifactId>
      <version>1.0-SNAPSHOT</version>
      <type>pom</type>
    </dependency>

    <!-- REST client for Monitoring -->
    <dependency>
      <groupId>eu.optimis.mi.restclient</groupId>
      <artifactId>RESTClient</artifactId>
      <version>0.0.1-SNAPSHOT</version>
    </dependency>

   <!-- REST client for FTE -->
    <dependency>
      <groupId>eu.optimis</groupId>
      <artifactId>FaultToleranceEngineRESTClient</artifactId>
```

```
    <version>1.0-SNAPSHOT</version>
</dependency>
<!-- Hibernate framework -->
<dependency>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>hibernate3-maven-plugin</artifactId>
    <version>2.2</version>
</dependency>

<!-- Hibernate library dependecy start -->
<dependency>
    <groupId>dom4j</groupId>
    <artifactId>dom4j</artifactId>
    <version>1.6.1</version>
</dependency>
<dependency>
    <groupId>commons-logging</groupId>
    <artifactId>commons-logging</artifactId>
    <version>1.1.1</version>
</dependency>
<dependency>
    <groupId>commons-collections</groupId>
    <artifactId>commons-collections</artifactId>
    <version>3.2.1</version>
</dependency>
<dependency>
    <groupId>cglib</groupId>
    <artifactId>cglib</artifactId>
    <version>2.2</version>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-log4j12</artifactId>
    <version>1.5.2</version>
    <type>jar</type>
    <scope>compile</scope>
</dependency>
<!-- Hibernate library dependecy end -->

<dependency>
```

```
    <groupId>javax.persistence</groupId>
    <artifactId>persistence-api</artifactId>
    <version>1.0</version>
</dependency>
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.14</version>
</dependency>
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.6</version>
</dependency>
<dependency>
    <groupId>hsqldb</groupId>
    <artifactId>hsqldb</artifactId>
    <version>1.8.0.7</version>
</dependency>
<dependency>
    <groupId>com.ssfnet</groupId>
    <artifactId>cernlite-api</artifactId>
    <version>1.0.0</version>
</dependency>
 <dependency>
    <groupId>eu.optimis.servicemanifest</groupId>
    <artifactId>service-manifest-api</artifactId>
    <version>1.0.6-SNAPSHOT</version>
</dependency>
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-math</artifactId>
    <version>2.0</version>
</dependency>
```

```
</dependencies>
```

The dependencies listed in Table 2 are included the pom file of the module maintained by the Apache Maven 3.0.3 automatically.

3.5 Installation Instructions

- SVN path: http://pandora.atosorigin.es/svn/optimis/branches/OptimisY3/InfrastructureProviderRiskAssessmentTool
- The IPRAT Client API competent (as a jar file) is maintained by the maven development environment; the following dependency should be included into the maven pom.xml file.

```
<dependency>
        <groupId>eu.optimis</groupId>
        <artifactId>InfrastructureProviderRiskAssessmentToolRESTClient</artifactId>
        <version>1.0-SNAPSHOT</version>
</dependency>
```

Please note IPRAT API is part of the TRECCommonAIPIP and can be used with TRECCommonAPIIP together:

```
<dependency>
        <groupId>eu.optimis.trec.common</groupId>
        <artifactId>TRECCommonAPIIP</artifactId>
        <version>3.0-SNAPSHOT </version>
 </dependency>
```

- The IPRAT Web Service competent is maintained by the maven development environment; the following dependency should be included into the maven pom.xml file.

```
<dependency>
        <groupId>eu.optimis</groupId>
        <artifactId> InfrastructureProviderRiskAssessmentToolRESTService </artifactId>
        <version>1.0-SNAPSHOT</version>
 </dependency>
```

The IPRAT Web Service competent can be built into a WAR file and deployed into the Tomcat web application container (e.g. /usr/local/apache-tomcat-6.0.32/webapps/).

3.6 Getting started

3.6.1 Using the Software

Once the IPRAT Web Service is deployed into the Tomcat, the IPRAT Client API will connect to it based on the configuration information defined by the optimis.properties file in /opt/optimis/etc.

3.6.2 Testing the Software

Java Unit Test cases are provided and can be executed in the maven environment by run the 'mvn test' command in the InfrastrucutreProviderRiskAssessmentTool directory.

3.6.3 Configuration

- log4jIPRA.properties in /opt/optimis/etc/InfrastructureProviderRiskAssessmentTool
- Log file: /opt/optimis/var/log/ServiceProviderRiskAssessmentTool/ InfrastructureProviderRiskAssessmentTool.log
- IPRAT's own configuration file is /opt/optimis/etc/InfrastructureProviderRiskAssessmentTool/ipraconfig.properties which include the shape and scale parameters of the weibull distribution:

  config.weibullpara1=0.3455

  config.weibullpara2=265000

3.7 FAQ