Institute of Systems Science

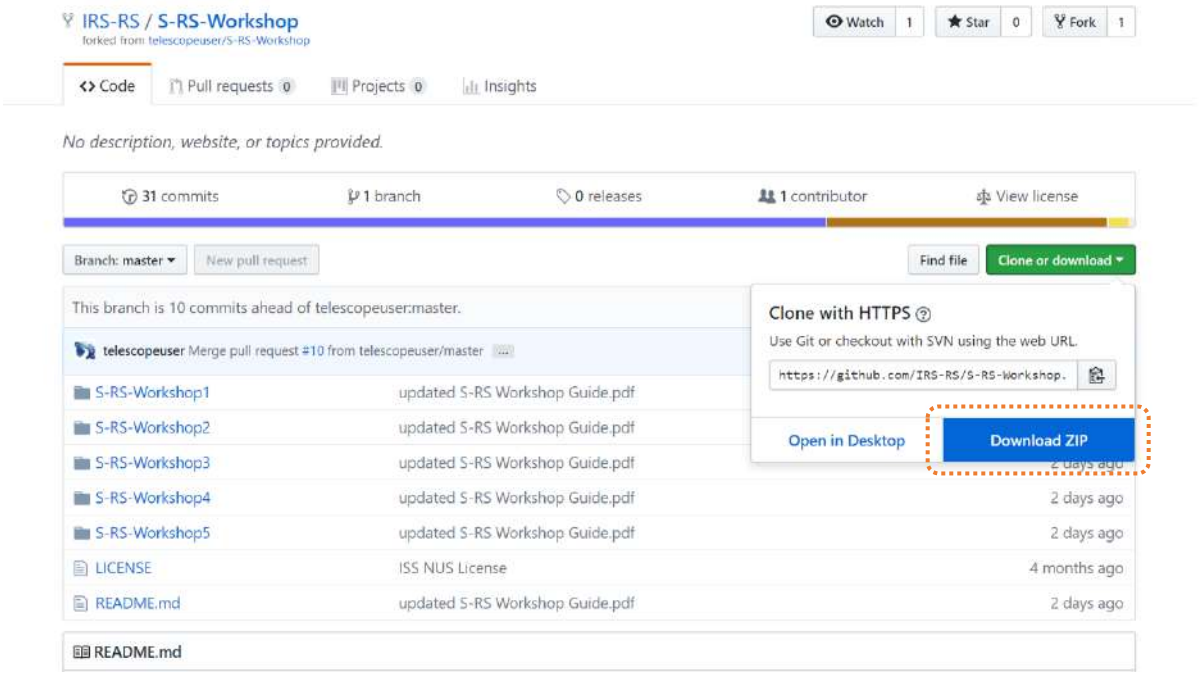
National University of Singapore

GRADUATE CERTIFICATE
INTELLIGENT REASONING SYSTEMS

Workshop Project Guide (2/5)

Subject: Reasoning Systems

Institute of Systems Science, 25 Heng Mui Keng Terrace, Singapore 119615



Workshops & References https://github.com/IRS-RS/S-RS-Workshop

Virtual Machine Workstation http://bit.ly/iss-vm

Table of Contents

1. Workshop 2 – Search Reasoning

WORKSHOP SEARCH REASONING

- **Cloud Balance Solver Deep Dive**
  - Cloud Balance Solver [Java IDE]
  - Cloud Balance Solver [KIE Workbench]
- **Cloud Balance Solver Enhancement**
  - GPU requirements; Data centre physical locations; Network latency, etc.

1.1. Cloud Balance Solver Deep Dive

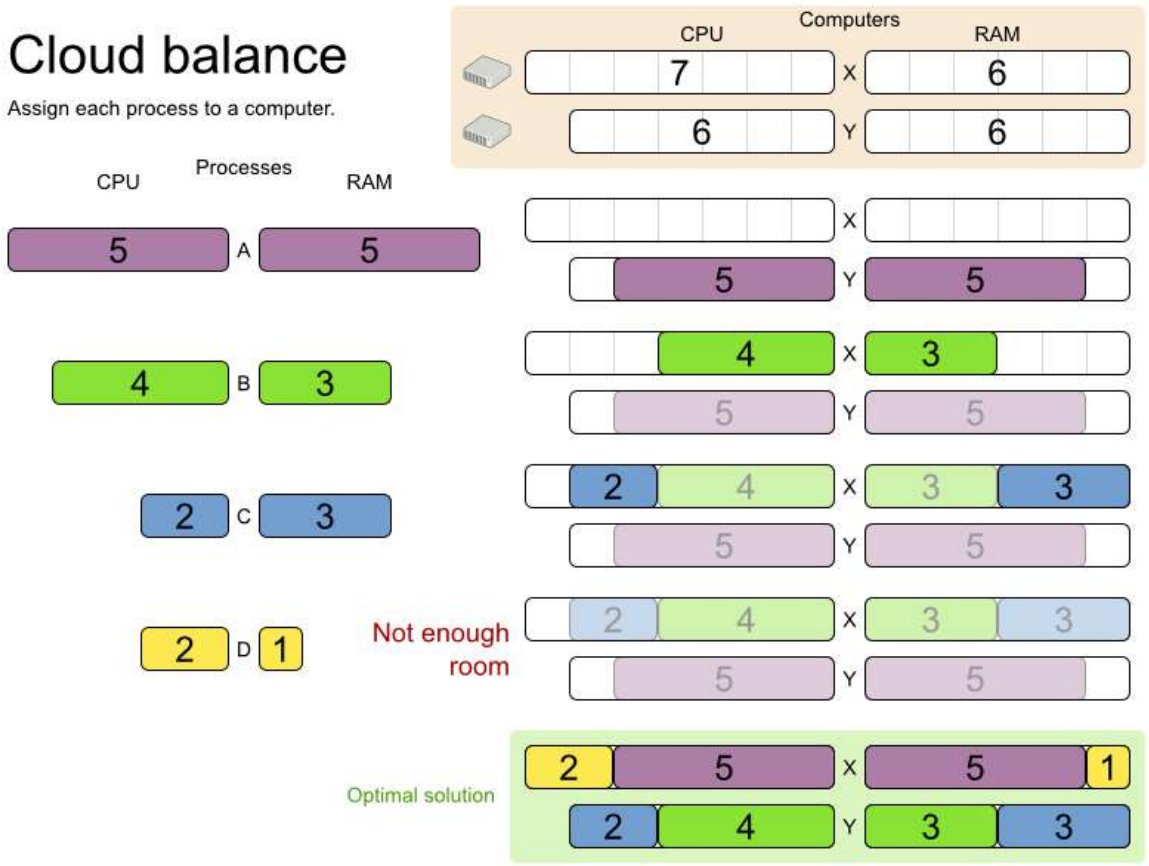
{ Objective }

Analyse and execute cloud computer balancing system/solver using both Eclipse IDE and KIE Workbench (KIE Server / RESTful API).

Decompose the end to end system solution, at system modelling level, to address the cloud balancing business resource optimization problem.

KIE OptaPlanner Deep Dive – Cloud Computer Balancing

- **Business Scenario / Problem Description**
- **A cloud service provider owns a number of cloud computers and needs to run a number of customers' processes on those computers. Assign each process to a computer.**

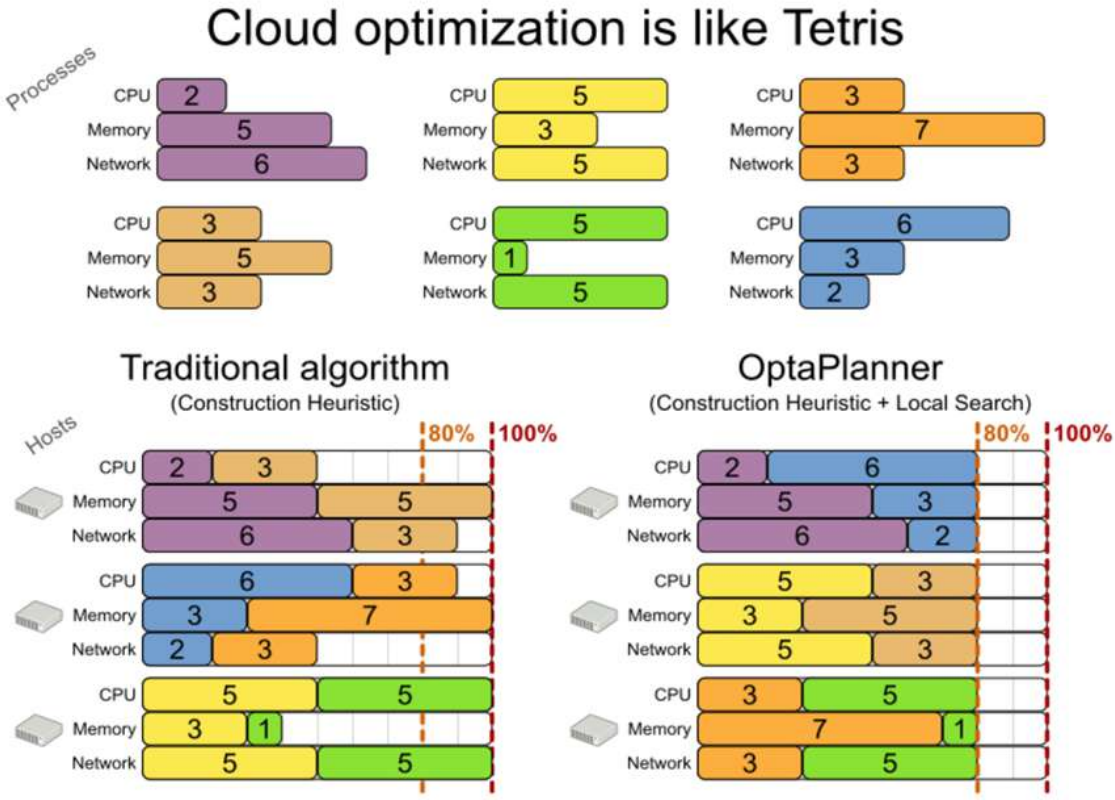


The following hard constraints must be fulfilled:

- Every computer must be able to handle the minimum hardware requirements of the sum of its processes:
  - **CPU capacity**: The CPU power of a computer must be at least the sum of the CPU power required by the processes assigned to that computer.
  - **Memory capacity**: The RAM memory of a computer must be at least the sum of the RAM memory required by the processes assigned to that computer.
  - **Network capacity**: The network bandwidth of a computer must be at least the sum of the network bandwidth required by the processes assigned to that computer.

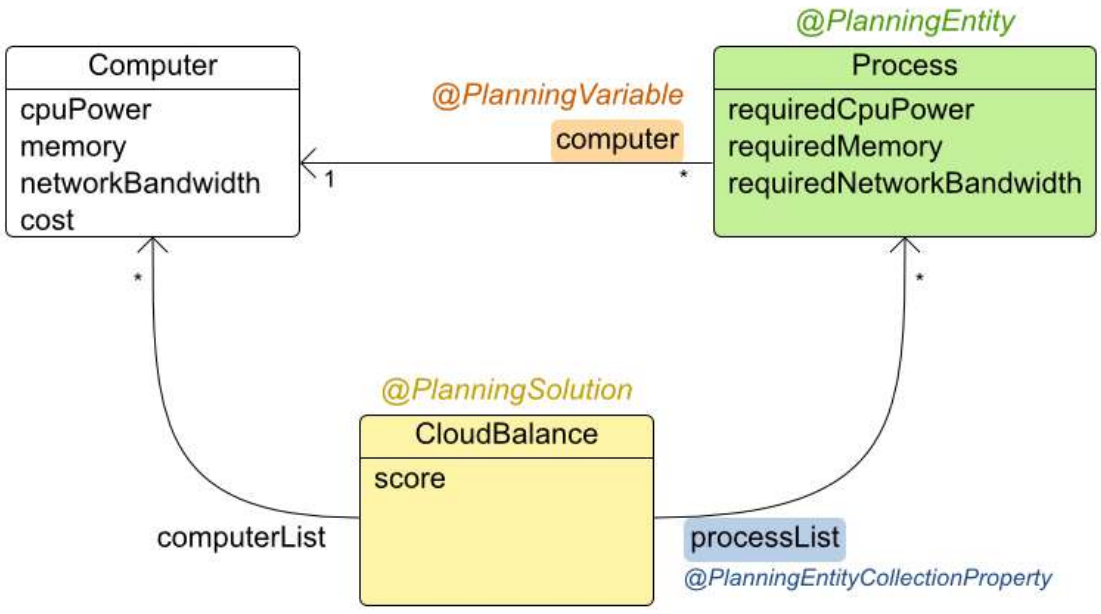
The following soft constraints should be optimized:

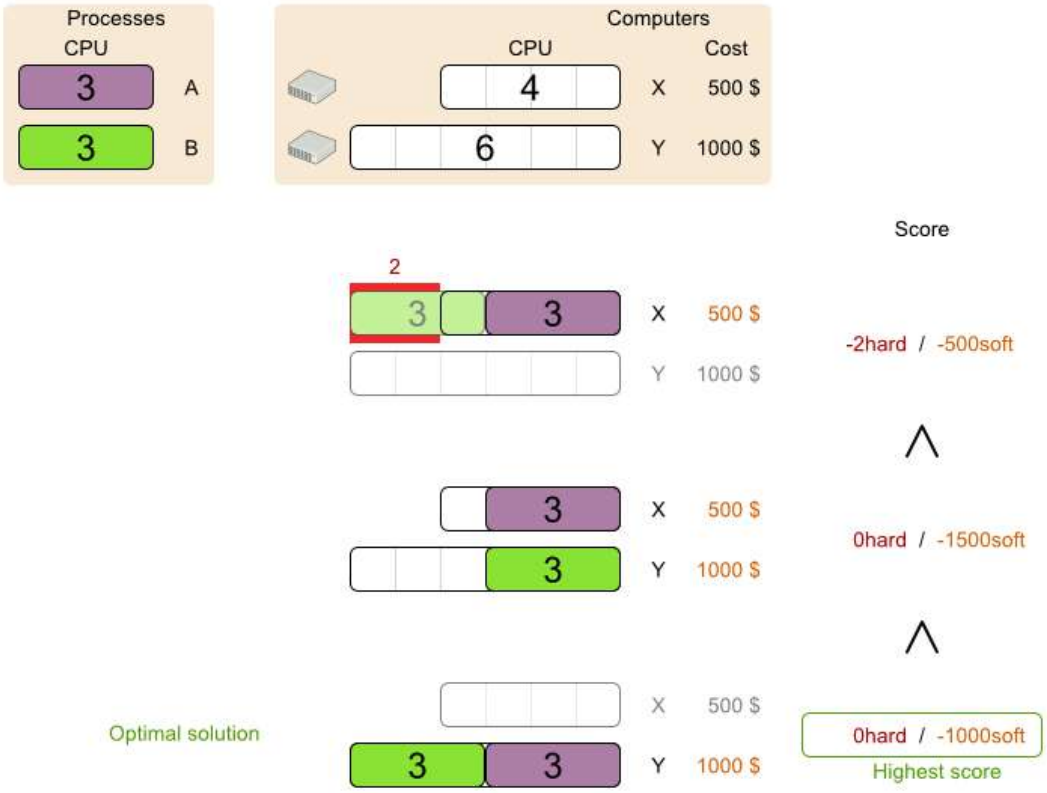
- Each computer that has one or more processes assigned, incurs a maintenance cost (which is fixed per computer).
  - **Cost**: Minimize the total maintenance cost.



- **Domain Modelling / Constraint Satisfaction**
  - To create a domain model, define all the objects that represent the input data for the problem. In this simple example, the objects are **processes** and **computers**.
  - A separate object (Solution Class) in the domain model must represent a full data set of problem, which contains the input data as well as a solution. In this example, this object holds a list of **computers** and a list of **processes**. Each process is assigned to a computer; the **distribution of processes between computers** is the **solution**.

Cloud balance class diagram





Drools score calculation

- Constraints in Drools Rule Language (DRL)
  - Declarative (like SQL, regular expression)
- Integration opportunities
  - Drools Workbench
  - Decision tables

Drools score calculation: facts

- Facts in DRL loaded from
  - `@ProblemFact(Collection)Property`
  - `@PlanningEntity(Collection)Property`

DRL hard constraint: CPU power

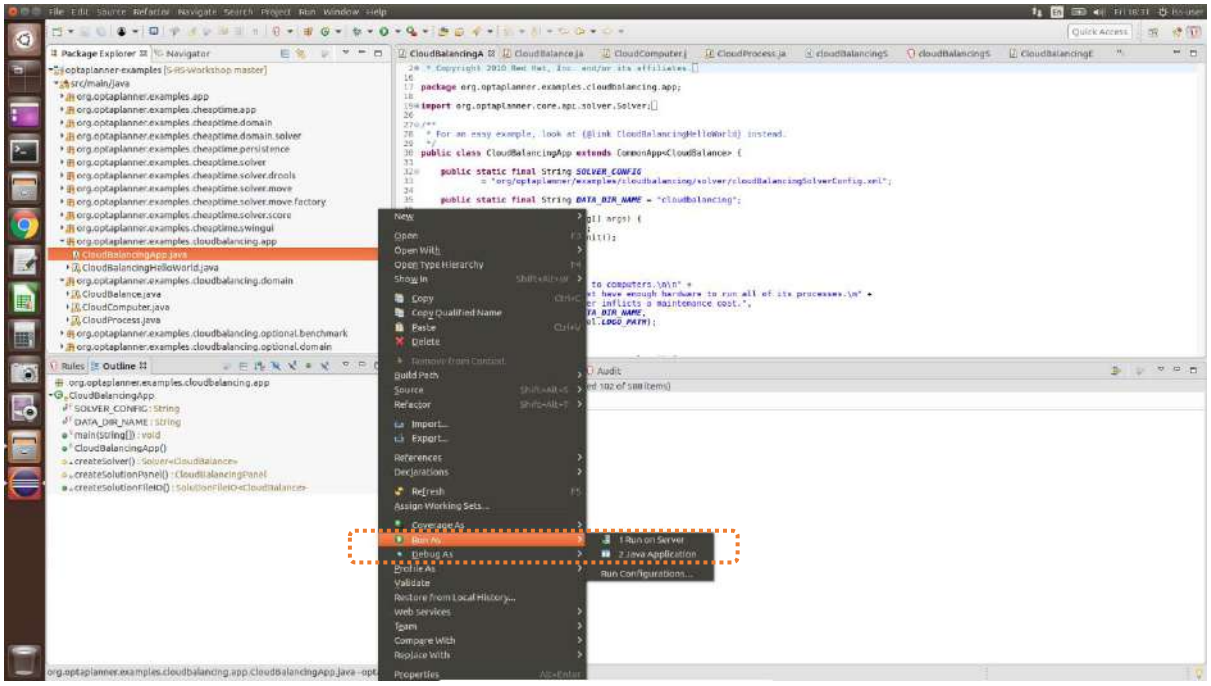
```
rule "requiredCpuPowerTotal"
  when
    // there is a computer
    $s : Computer($cpu : cpuPower)
    // with too little cpu for its processes
    accumulate(
      Process(computer == $s, $requiredCpu : requiredCpuPower);
      $total : sum($requiredCpu);
      $total > $cpu
    )
  then
    // lower hard score by the excessive CPU usage
    scoreHolder.addHardConstraintMatch(kcontext,
        $cpu - $total);
end
```

DRL soft constraint: computer cost

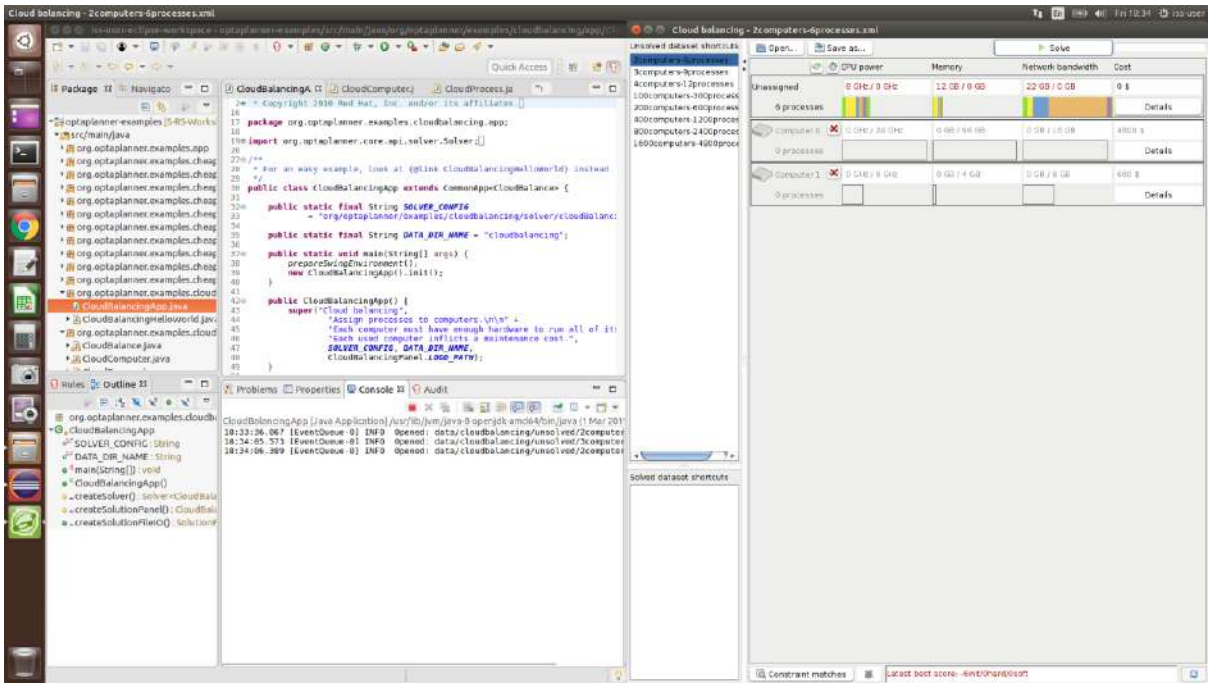
```
rule "computerCost"
  when
    // there is a computer
    $s : Computer($c : cost)
    // there is a processes on that computer
    exists Process(computer == $s)
  then
    // lower soft score by the maintenance cost
    scoreHolder.addSoftConstraintMatch(kcontext, - $c);
end
```

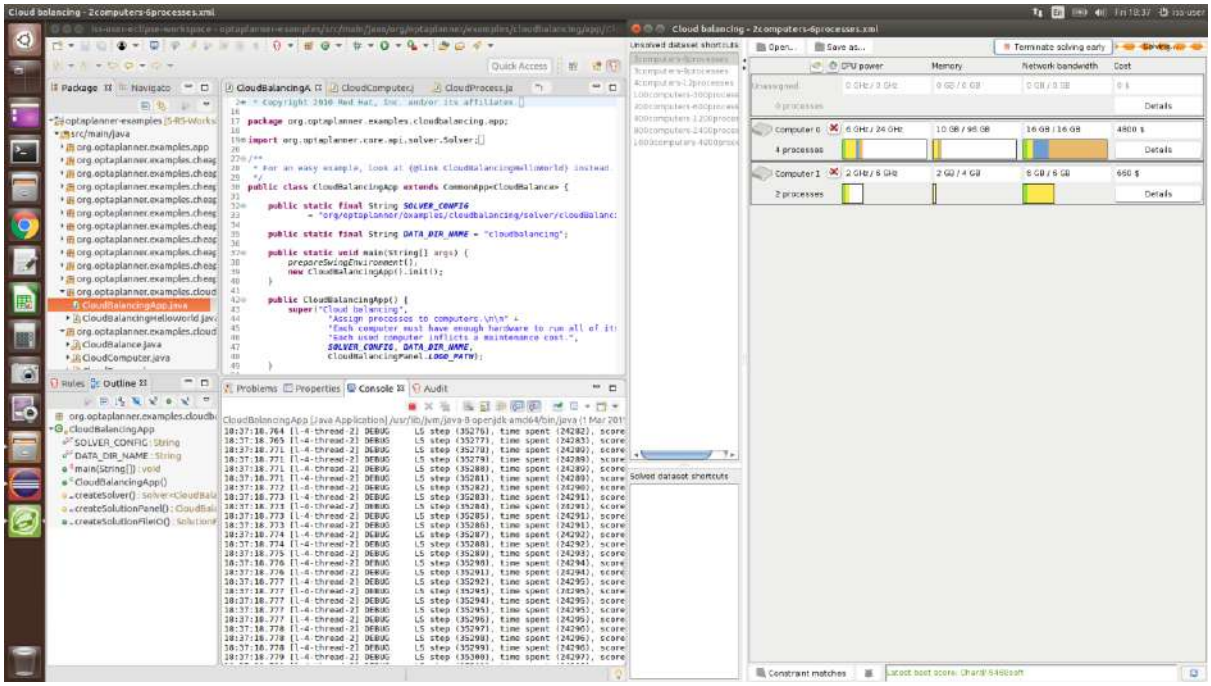
1.1.1. Cloud Balance Solver [Java IDE]

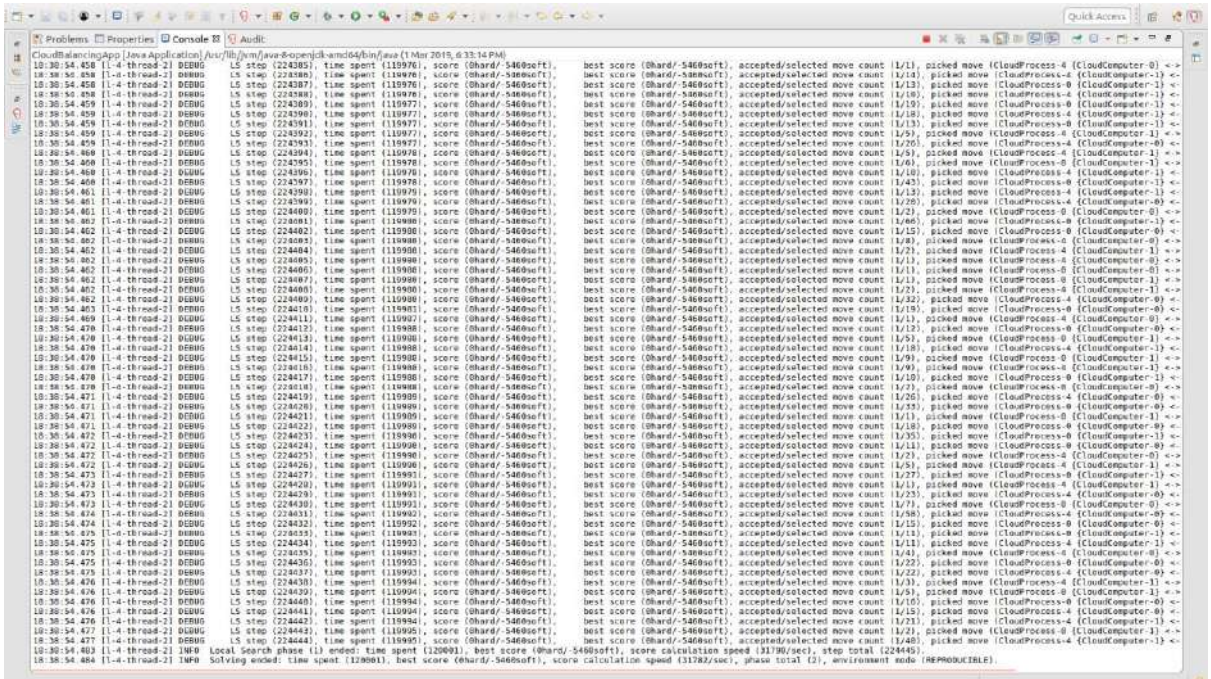
1) Expand project to ***CloudBalancingApp.java***; right click mouse to call menu; Select **Run As** : **2 Java Application**;



2) Select **2 computers – 6 processes**; Click **Solve**;

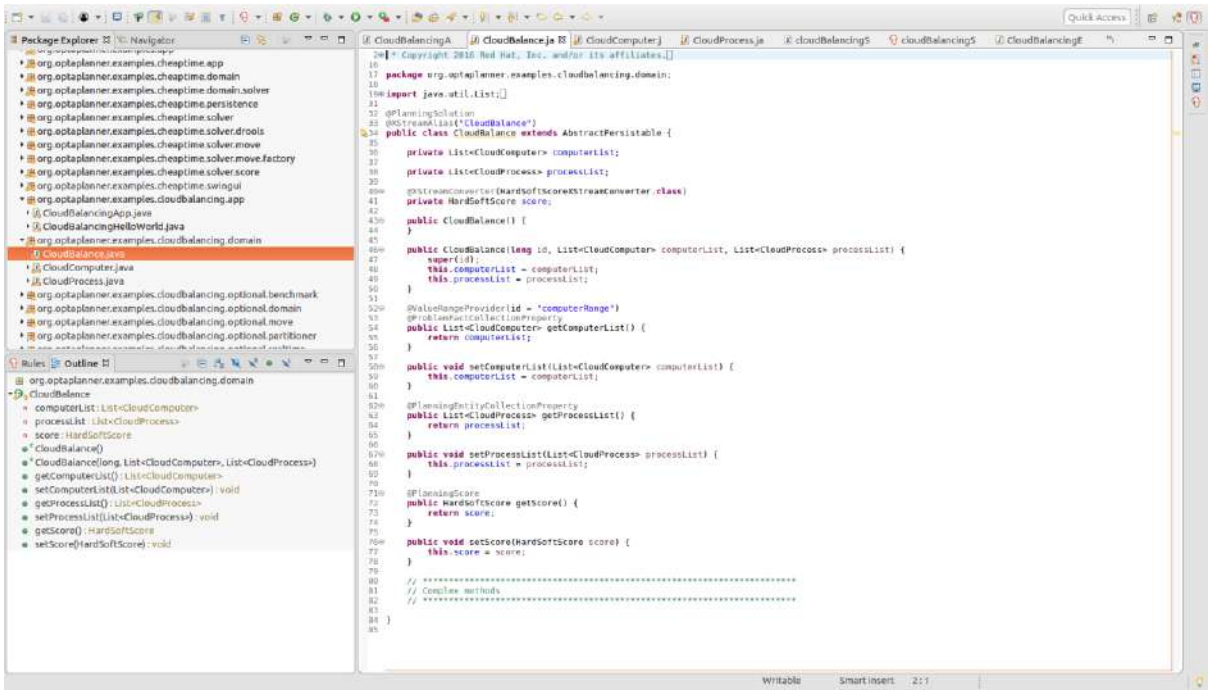




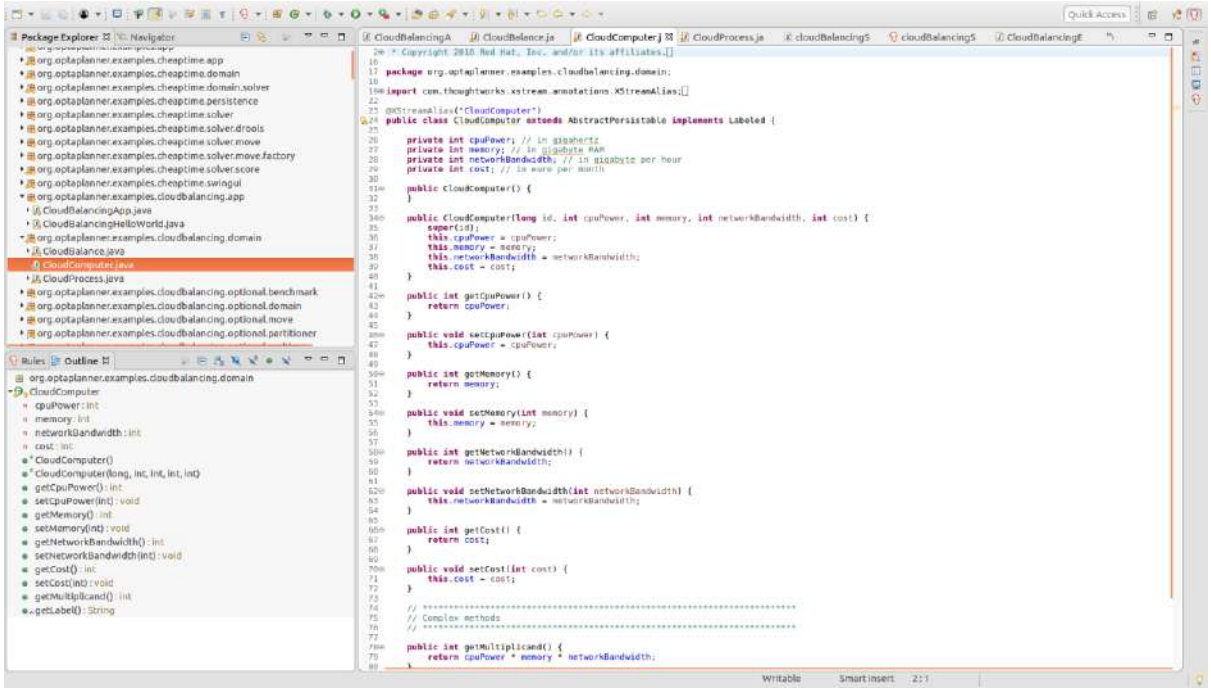


3) Review class diagram; Map against Java classes/objects

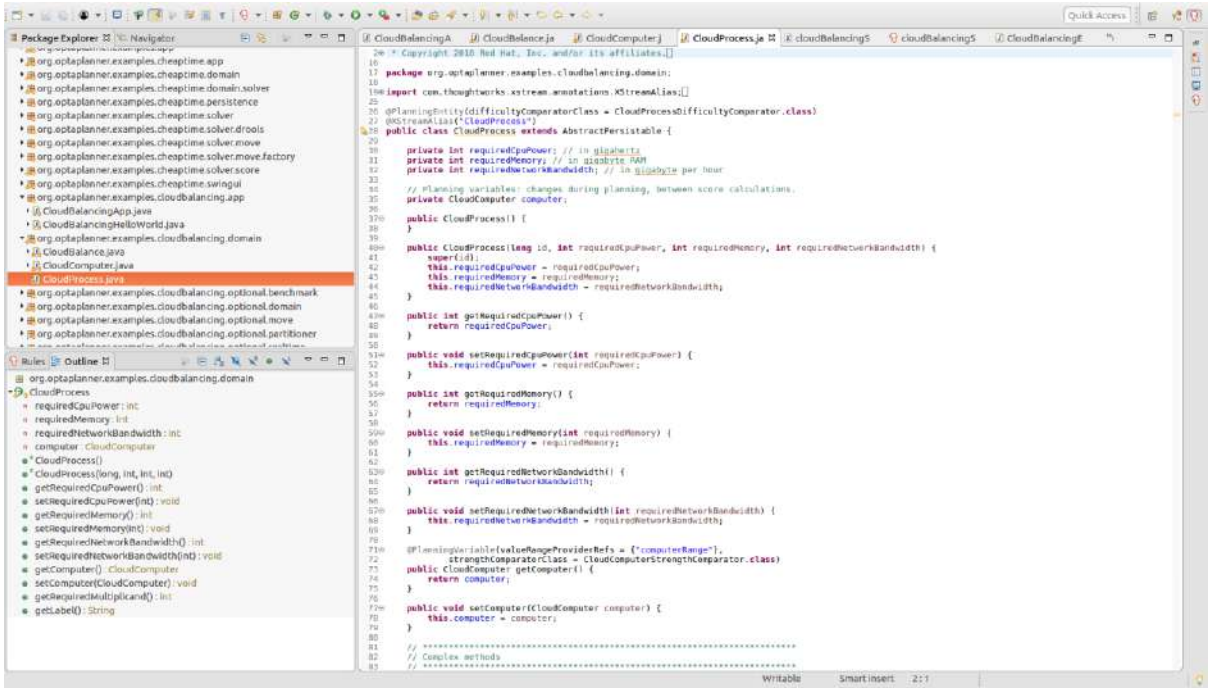
CloudBalance.java



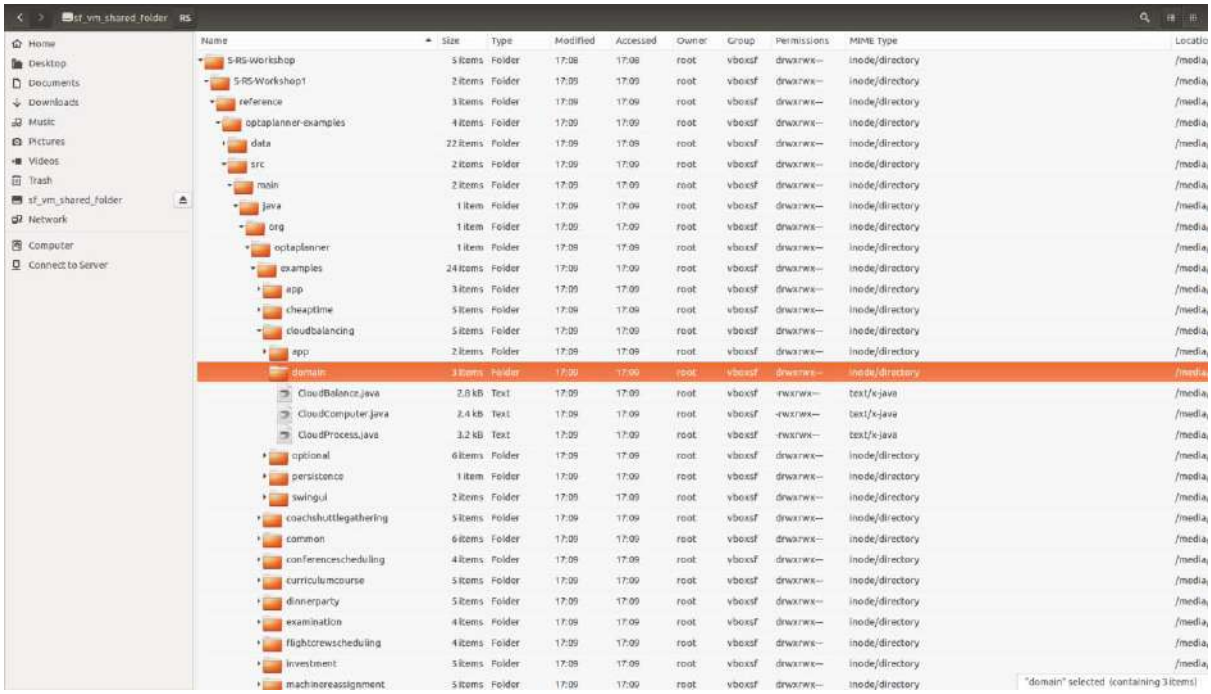
CloudComputer.java



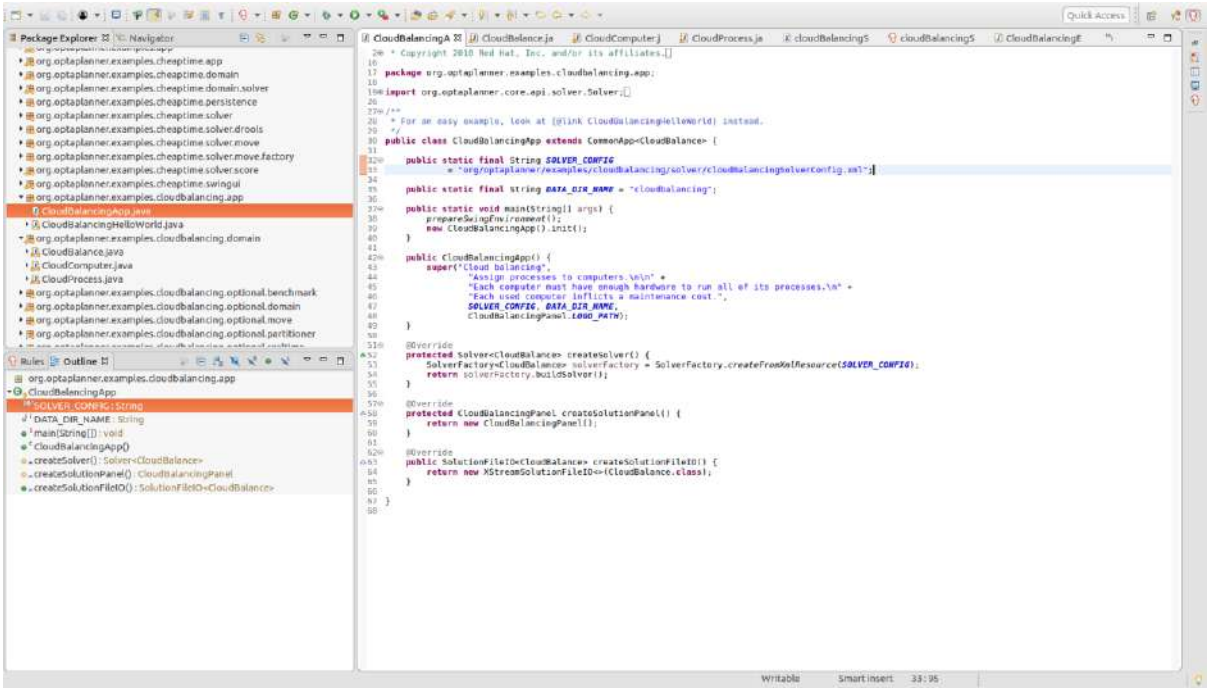
CloudProcess.java

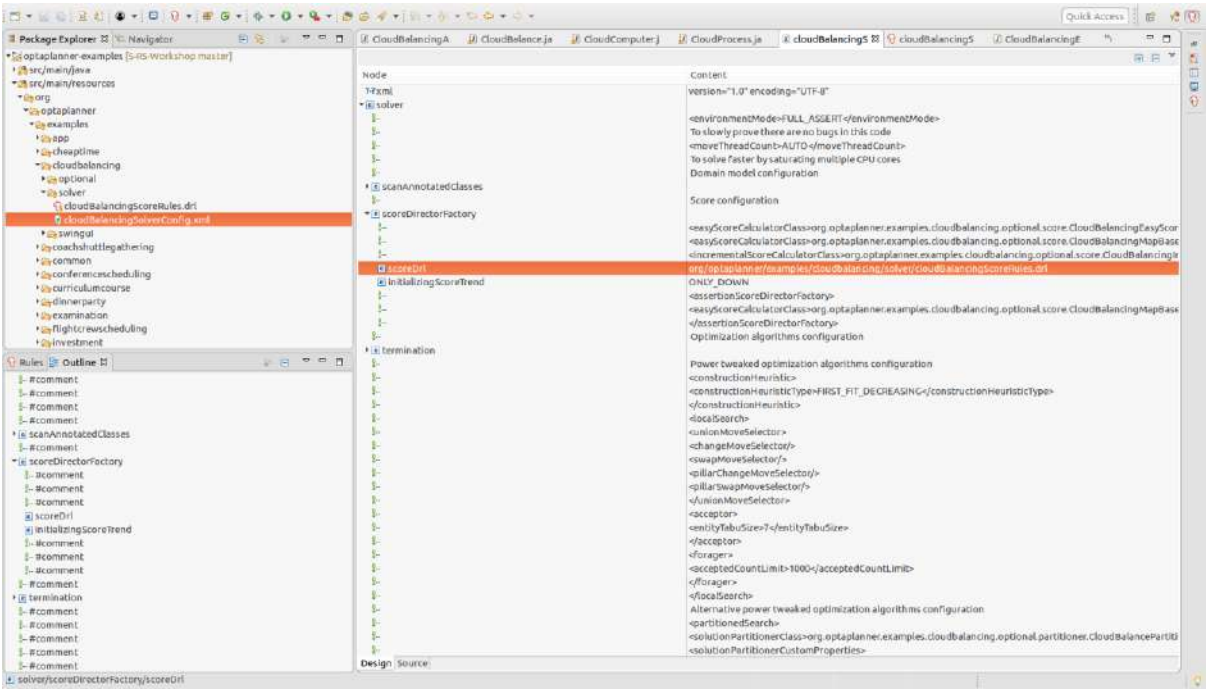


Physical file's location:

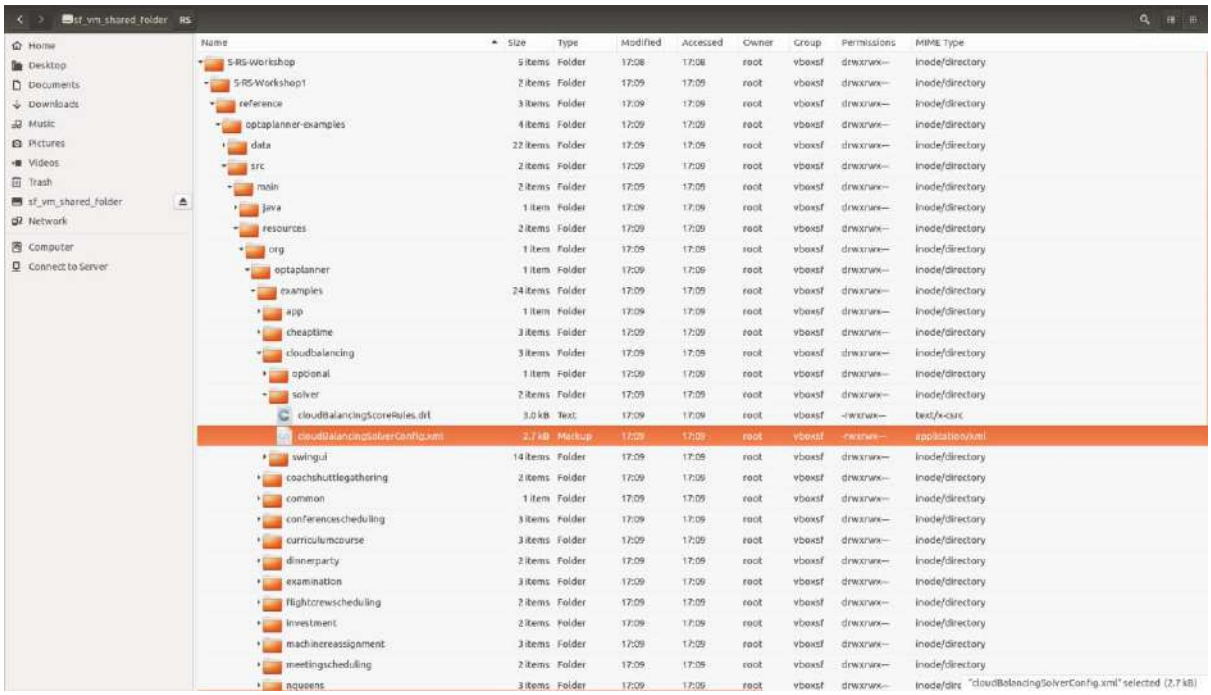


4) Review solver configuration file (XML): ***org/optaplanner/examples/cloudbalancing/solver/cloudBalancingSolverConfig.xml***



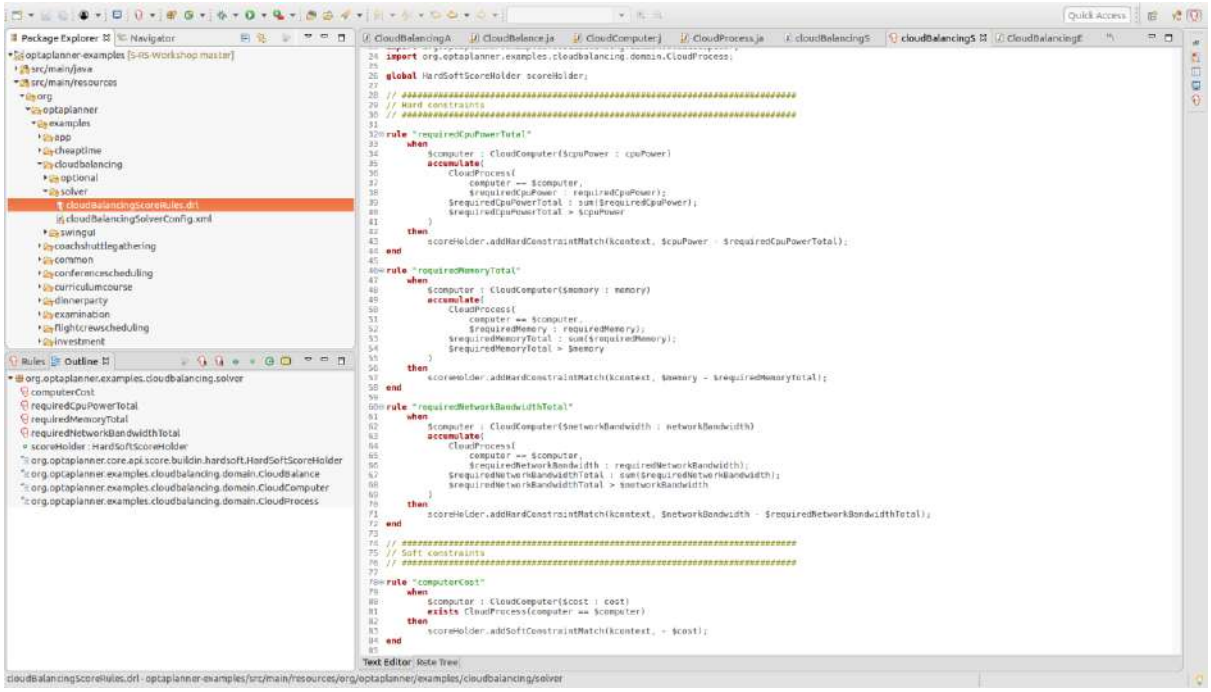


Physical file's location:

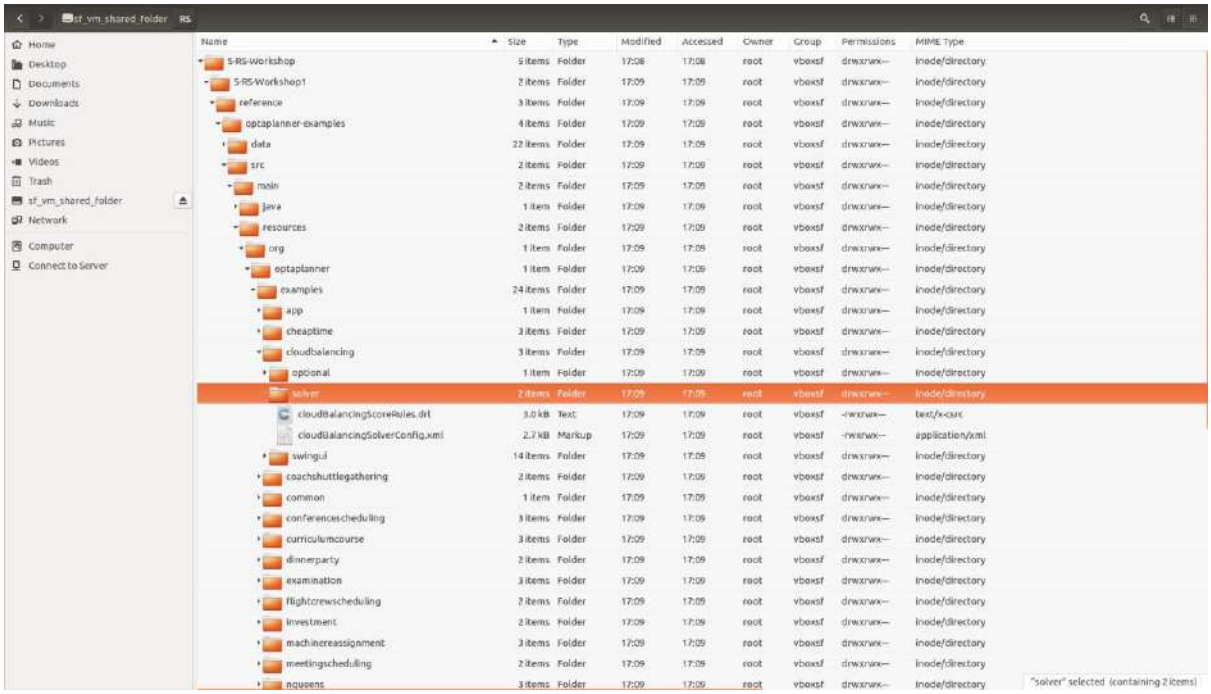


```
<?xml version="1.0" encoding="UTF-8"?>
<solver>
  <!--<environmentMode>FULL_ASSERT</environmentMode>--><!-- To slowly prove there are no bugs
in this code -->
  <!--<moveThreadCount>AUTO</moveThreadCount>--><!-- To solve faster by saturating multiple
CPU cores -->

  <!-- Domain model configuration -->
  <scanAnnotatedClasses>
    <packageInclude>org.optaplanner.examples.cloudbalancing</packageInclude>
  </scanAnnotatedClasses>

  <!-- Score configuration -->
  <scoreDirectorFactory>
    <!--
<easyScoreCalculatorClass>org.optaplanner.examples.cloudbalancing.optional.score.CloudBalancin
gEasyScoreCalculator</easyScoreCalculatorClass>-->
    <!--
<easyScoreCalculatorClass>org.optaplanner.examples.cloudbalancing.optional.score.CloudBalancin
gMapBasedEasyScoreCalculator</easyScoreCalculatorClass>-->
    <!--
<incrementalScoreCalculatorClass>org.optaplanner.examples.cloudbalancing.optional.score.CloudB
alancingIncrementalScoreCalculator</incrementalScoreCalculatorClass>-->

<scoreDrl>org/optaplanner/examples/cloudbalancing/solver/cloudBalancingScoreRules.drl</scoreDr
l>
    <initializingScoreTrend>ONLY_DOWN</initializingScoreTrend>
    <!--<assertionScoreDirectorFactory>-->
      <!--
<easyScoreCalculatorClass>org.optaplanner.examples.cloudbalancing.optional.score.CloudBalancin
gMapBasedEasyScoreCalculator</easyScoreCalculatorClass>-->
    <!--</assertionScoreDirectorFactory>-->
  </scoreDirectorFactory>

  <!-- Optimization algorithms configuration -->
  <termination>
    <minutesSpentLimit>2</minutesSpentLimit>
  </termination>

  <!-- Power tweaked optimization algorithms configuration -->
  <!--<constructionHeuristic>-->
    <!--<constructionHeuristicType>FIRST_FIT_DECREASING</constructionHeuristicType>-->
  <!--</constructionHeuristic>-->
  <!--<localSearch>-->
    <!--<unionMoveSelector>-->
      <!--<changeMoveSelector/>-->
      <!--<swapMoveSelector/>-->
      <!--<pillarChangeMoveSelector/>-->
      <!--<pillarSwapMoveSelector/>-->
    <!--</unionMoveSelector>-->
    <!--<acceptor>-->
      <!--<entityTabuSize>7</entityTabuSize>-->
    <!--</acceptor>-->
    <!--<forager>-->
      <!--<acceptedCountLimit>1000</acceptedCountLimit>-->
    <!--</forager>-->
  <!--</localSearch>-->

  <!-- Alternative power tweaked optimization algorithms configuration -->
  <!--<partitionedSearch>-->
    <!--
<solutionPartitionerClass>org.optaplanner.examples.cloudbalancing.optional.partitioner.CloudBa
lancePartitioner</solutionPartitionerClass>-->
    <!--<solutionPartitionerCustomProperties>-->
      <!--<partCount>4</partCount>-->
      <!--<minimumProcessListSize>300</minimumProcessListSize>-->
    <!--</solutionPartitionerCustomProperties>-->
  <!--</partitionedSearch>-->
</solver>
```

5) Review solver score calculation file (Drools Rule): ***org/optaplanner/examples/cloudbalancing/solver/cloudBalancingScoreRules.drl***



Physical file's location:

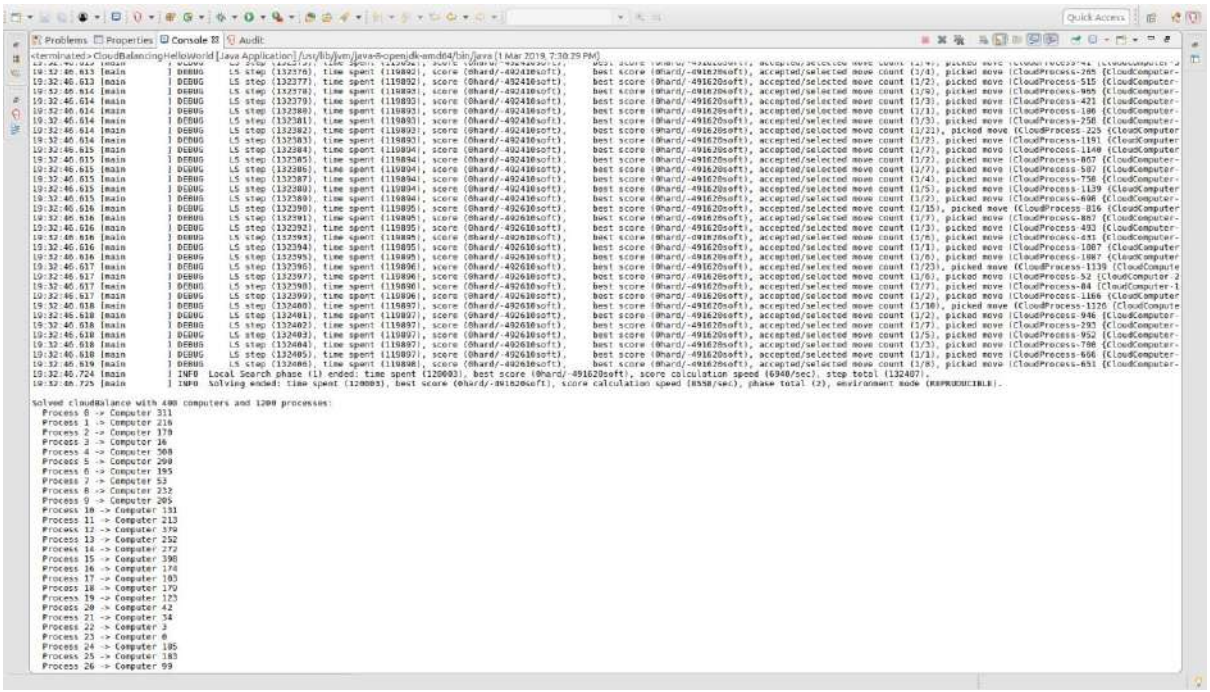


```
/*
 * Copyright 2010 Red Hat, Inc. and/or its affiliates.
 *
 */

package org.optaplanner.examples.cloudbalancing.solver;
    dialect "java"

import org.optaplanner.core.api.score.buildin.hardsoft.HardSoftScoreHolder;

import org.optaplanner.examples.cloudbalancing.domain.CloudBalance;
import org.optaplanner.examples.cloudbalancing.domain.CloudComputer;
import org.optaplanner.examples.cloudbalancing.domain.CloudProcess;

global HardSoftScoreHolder scoreHolder;

// ############################################################################
// Hard constraints
// ############################################################################

rule "requiredCpuPowerTotal"
    when
        $computer : CloudComputer($cpuPower : cpuPower)
        accumulate(
            CloudProcess(
                computer == $computer,
                $requiredCpuPower : requiredCpuPower);
            $requiredCpuPowerTotal : sum($requiredCpuPower);
            $requiredCpuPowerTotal > $cpuPower
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $cpuPower - $requiredCpuPowerTotal);
end

rule "requiredMemoryTotal"
    when
        $computer : CloudComputer($memory : memory)
        accumulate(
            CloudProcess(
                computer == $computer,
                $requiredMemory : requiredMemory);
            $requiredMemoryTotal : sum($requiredMemory);
            $requiredMemoryTotal > $memory
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $memory - $requiredMemoryTotal);
end

rule "requiredNetworkBandwidthTotal"
    when
        $computer : CloudComputer($networkBandwidth : networkBandwidth)
        accumulate(
            CloudProcess(
                computer == $computer,
                $requiredNetworkBandwidth : requiredNetworkBandwidth);
            $requiredNetworkBandwidthTotal : sum($requiredNetworkBandwidth);
            $requiredNetworkBandwidthTotal > $networkBandwidth
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $networkBandwidth -
$requiredNetworkBandwidthTotal);
end

// ############################################################################
// Soft constraints
// ############################################################################

rule "computerCost"
    when
        $computer : CloudComputer($cost : cost)
        exists CloudProcess(computer == $computer)
    then
        scoreHolder.addSoftConstraintMatch(kcontext, - $cost);
end
```

6) [Optional] { Objective } To compare performance (speed difference) in getting a solution using different score calculation methods, e.g. Drools Rule, Easy Java, Incremental Java.

7) [Optional] Run the solver to obtain performance using Drool Rule. You can run Java program CloudBalanceHelloWorld.java in console mode, to record the **score calculation speed**.

score calculation speed using **Drools Rule**



```
19:32:46.618 [main         ] DEBUG     LS step (132405), time spent
(119897), score (0hard/-492610soft),     best score (0hard/-491620soft),
accepted/selected move count (1/1), picked move (CloudProcess-666
{CloudComputer-87} <-> CloudProcess-96 {CloudComputer-355}).

19:32:46.619 [main         ] DEBUG     LS step (132406), time spent
(119898), score (0hard/-492610soft),     best score (0hard/-491620soft),
accepted/selected move count (1/8), picked move (CloudProcess-651
{CloudComputer-263} <-> CloudProcess-87 {CloudComputer-369}).

19:32:46.724 [main         ] INFO  Local Search phase (1) ended: time spent
(120003), best score (0hard/-491620soft), score calculation speed
(6940/sec), step total (132407).

19:32:46.725 [main         ] INFO  Solving ended: time spent (120003), best
score (0hard/-491620soft), score calculation speed (8558/sec), phase total
(2), environment mode (REPRODUCIBLE).
```

8) [Optional] Run the solver to obtain performance using Easy Java. You can run Java program CloudBalanceHelloWorld.java in console mode, to record the **score calculation speed**.

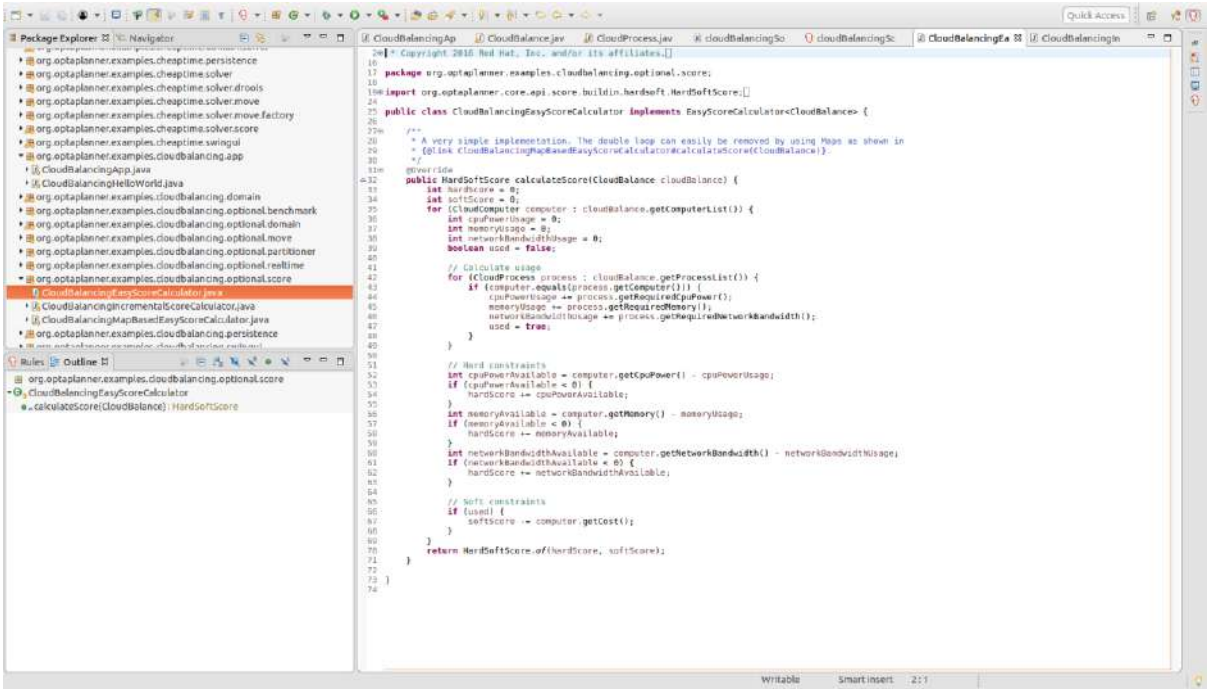
Update **SOLVER_CONFIG** file ***cloudBalancingSolverConfig.xml***

From using Drools Rule:

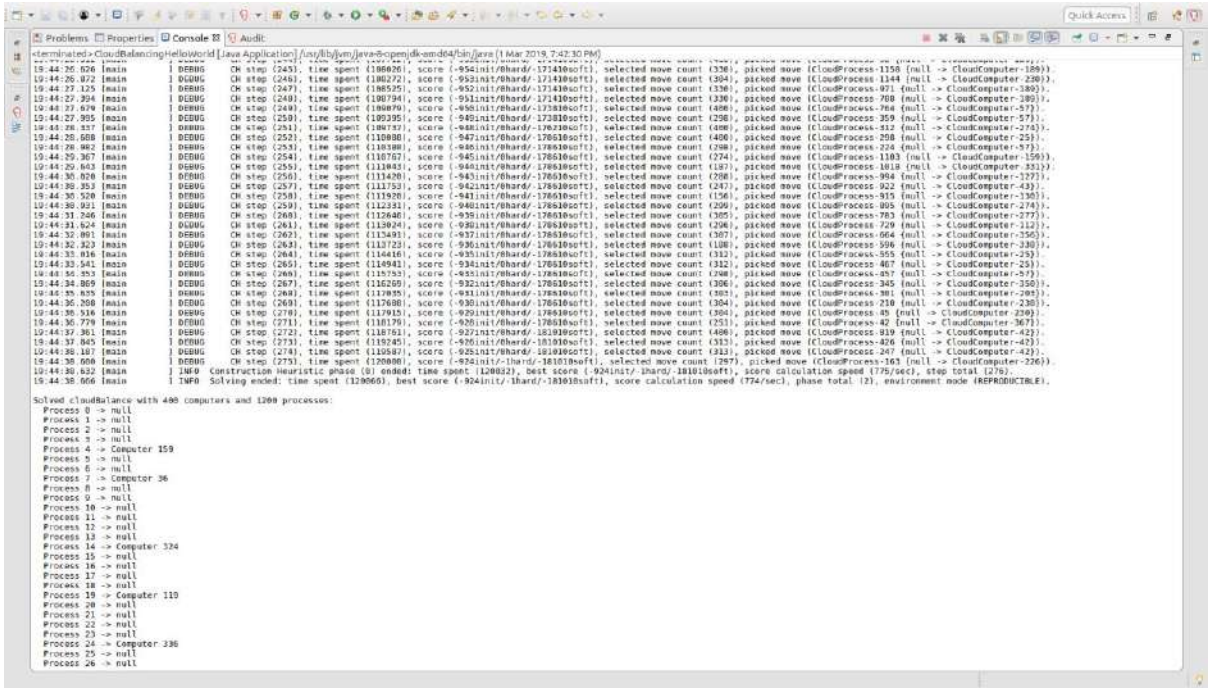
<scoreDrl>*org/optaplanner/examples/cloudbalancing/solver/cloudBalancingScoreRules.drl*</scoreDrl>

To using Easy Java:

<easyScoreCalculatorClass>*org.optaplanner.examples.cloudbalancing.optional.score.CloudBalancingEasyScoreCalculator*</easyScoreCalculatorClass>



score calculation speed using **Easy Java**



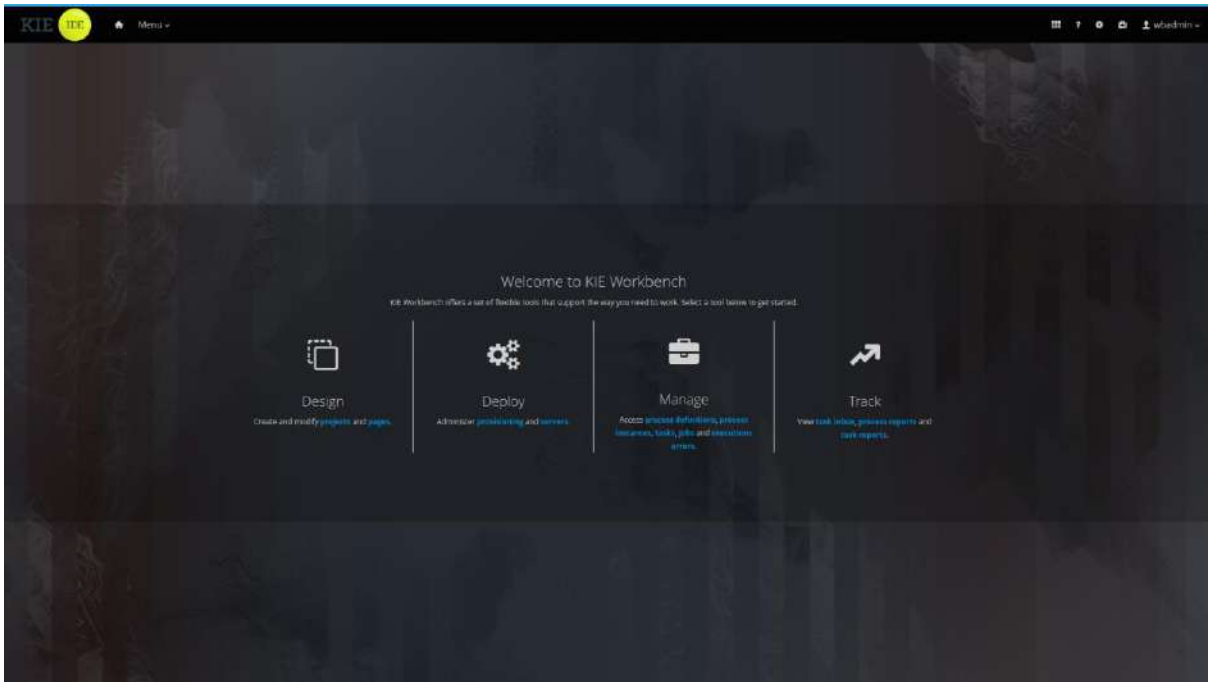
```
19:44:38.187 [main         ] DEBUG     CH step (274), time spent (119587),
score (-925init/0hard/-181010soft), selected move count (313), picked move
(CloudProcess-247 {null -> CloudComputer-42}).

19:44:38.600 [main         ] DEBUG     CH step (275), time spent (120000),
score (-924init/-1hard/-181010soft), selected move count (297), picked move
(CloudProcess-163 {null -> CloudComputer-226}).

19:44:38.632 [main         ] INFO  Construction Heuristic phase (0) ended:
time spent (120032), best score (-924init/-1hard/-181010soft), score
calculation speed (775/sec), step total (276).

19:44:38.666 [main         ] INFO  Solving ended: time spent (120066), best
score (-924init/-1hard/-181010soft), score calculation speed (774/sec),
phase total (2), environment mode (REPRODUCIBLE).
```

9) [Optional] { Challenge } score calculation speed using Incremental Java;

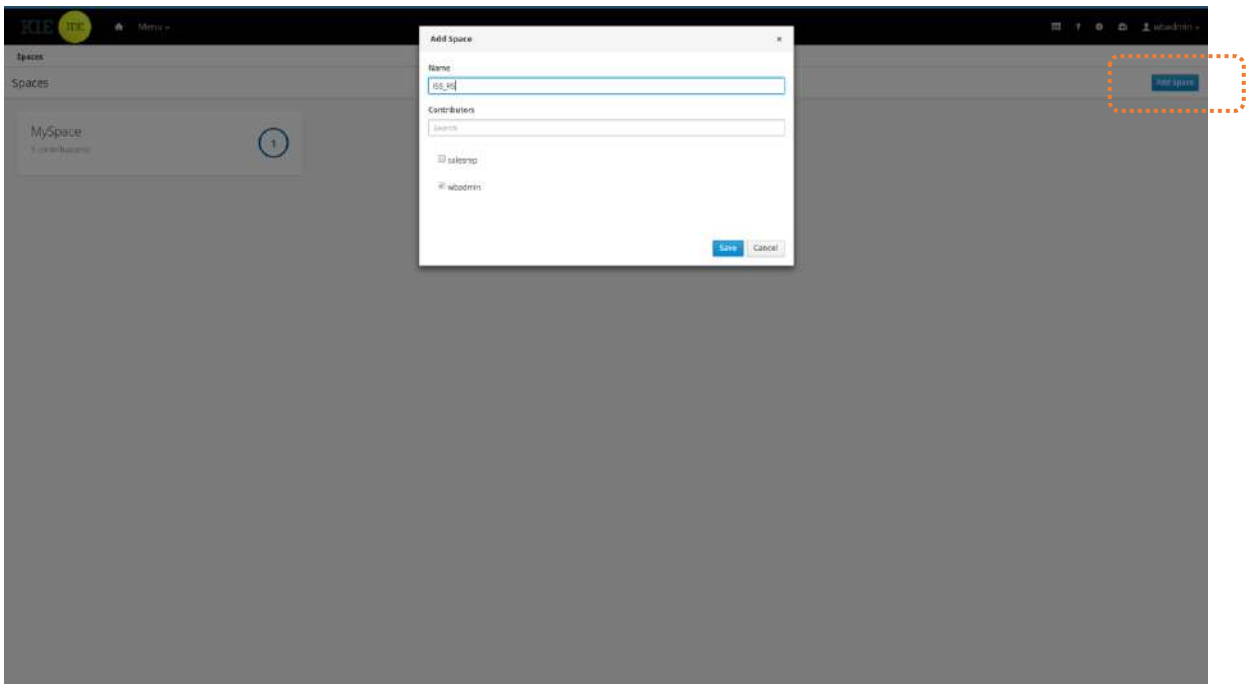
1.1.2. Cloud Balance Solver [KIE Workbench] v1.0.0

1) Install RESTful API tool: Postman
   Refer to Annex 6;

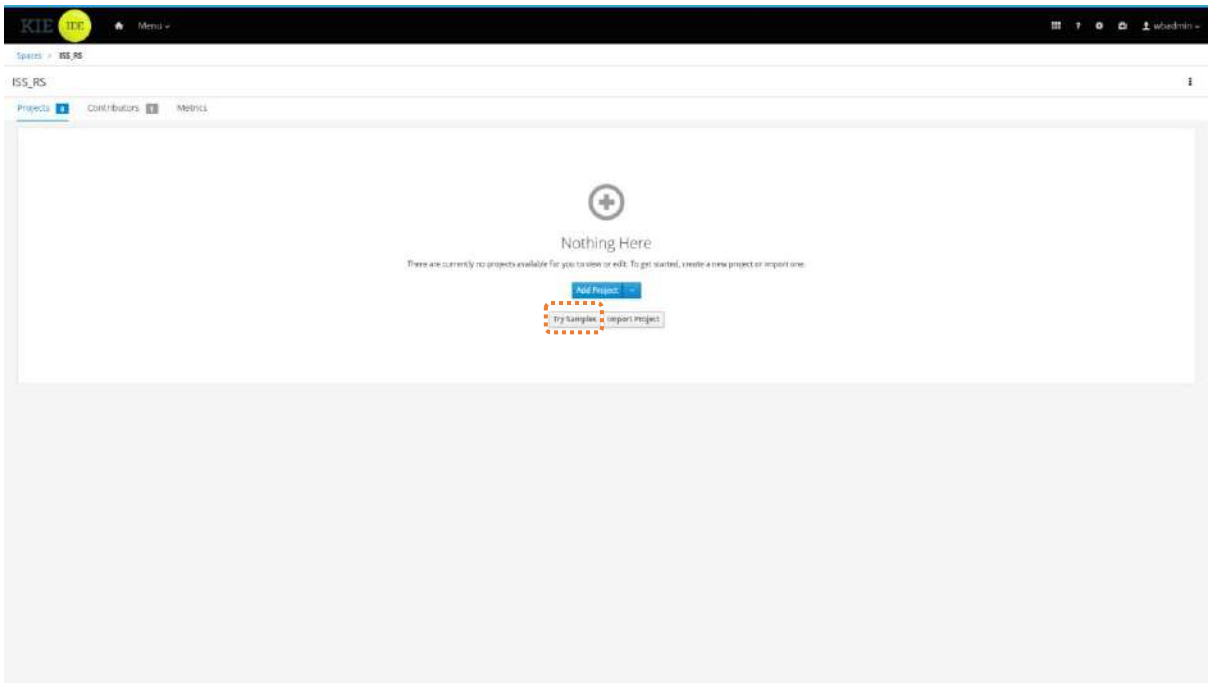
2) Start KIE Workbench



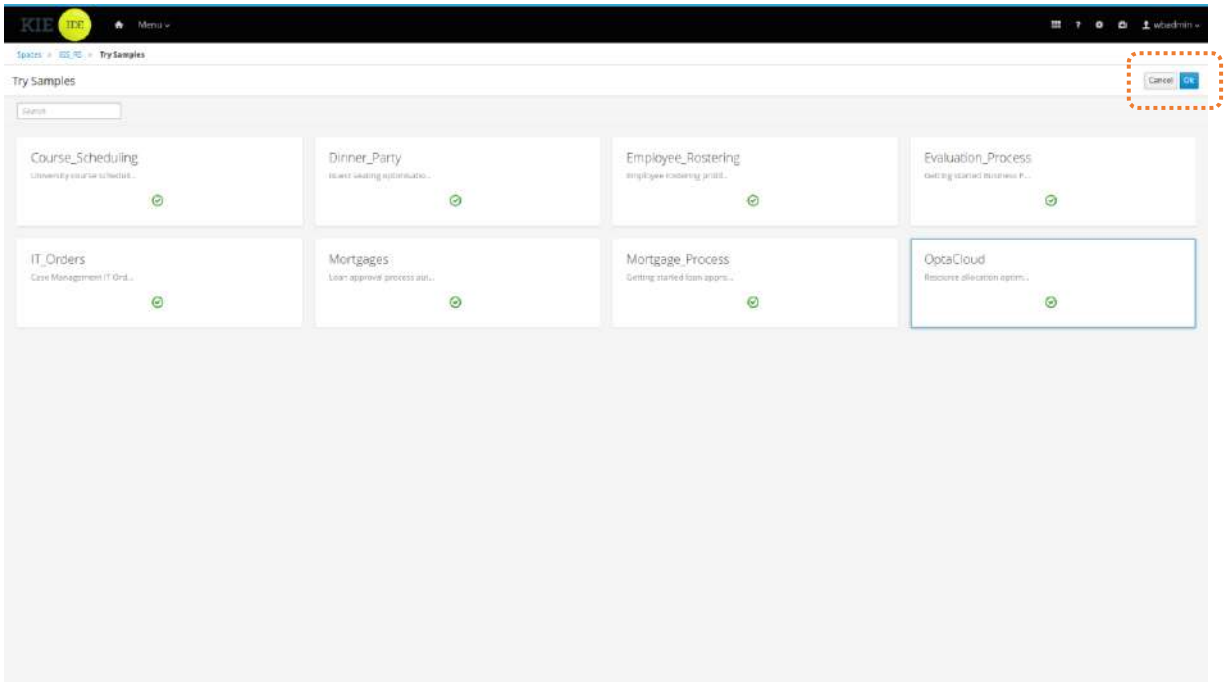
3) Create workspace: ***ISS_RS***

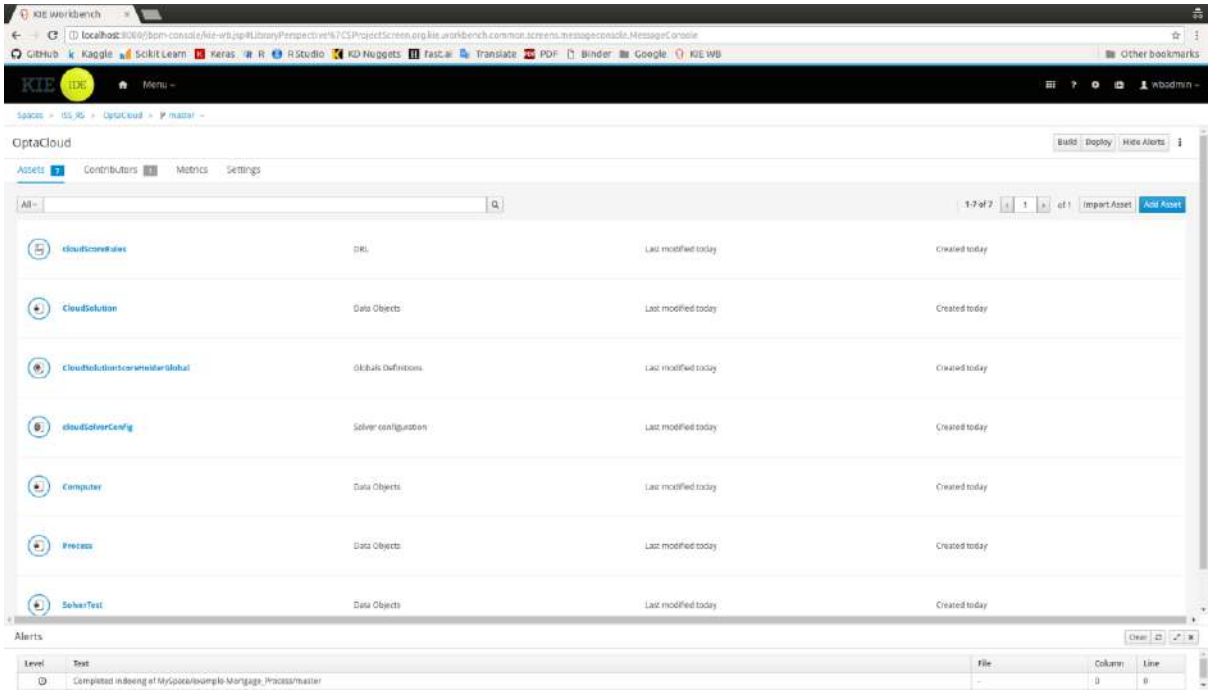


4) Click **Try Samples**



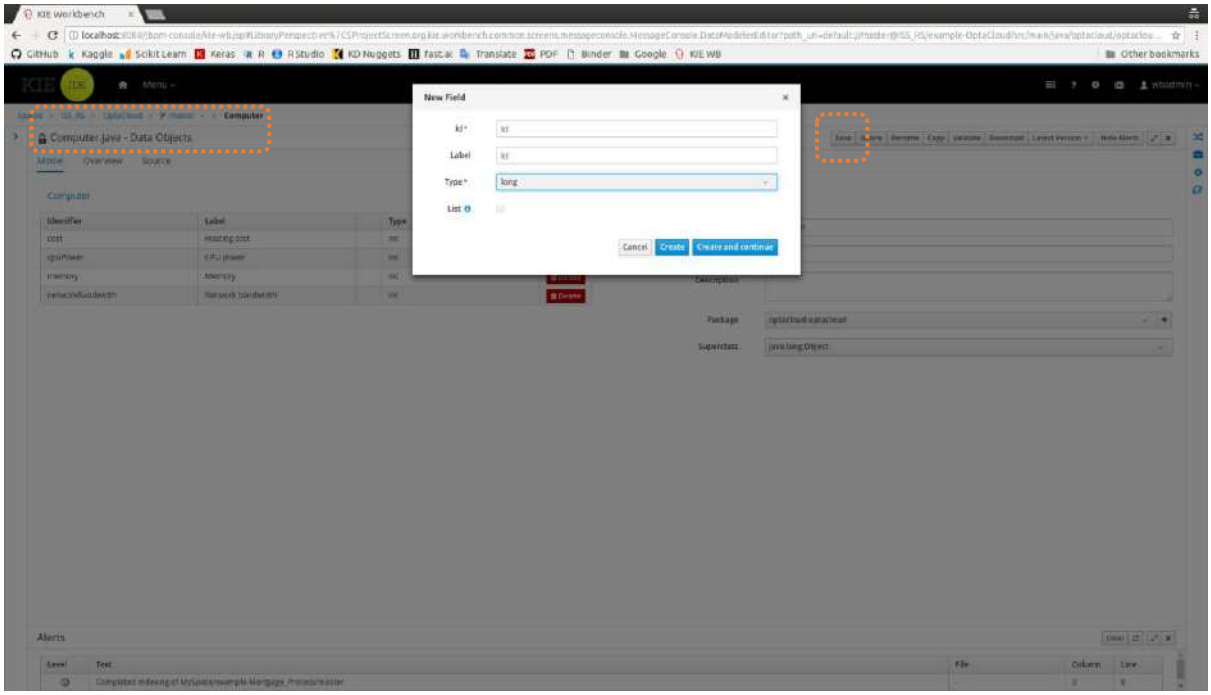
5) Select ***OptaCloud***; Click **OK**

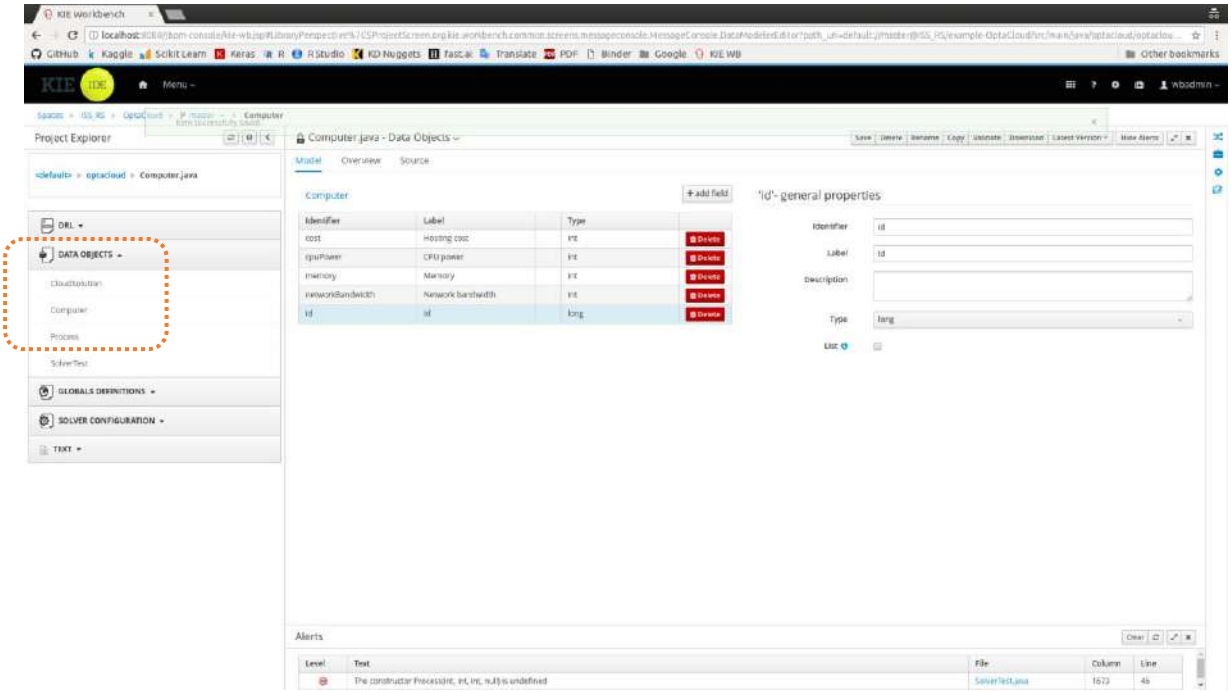


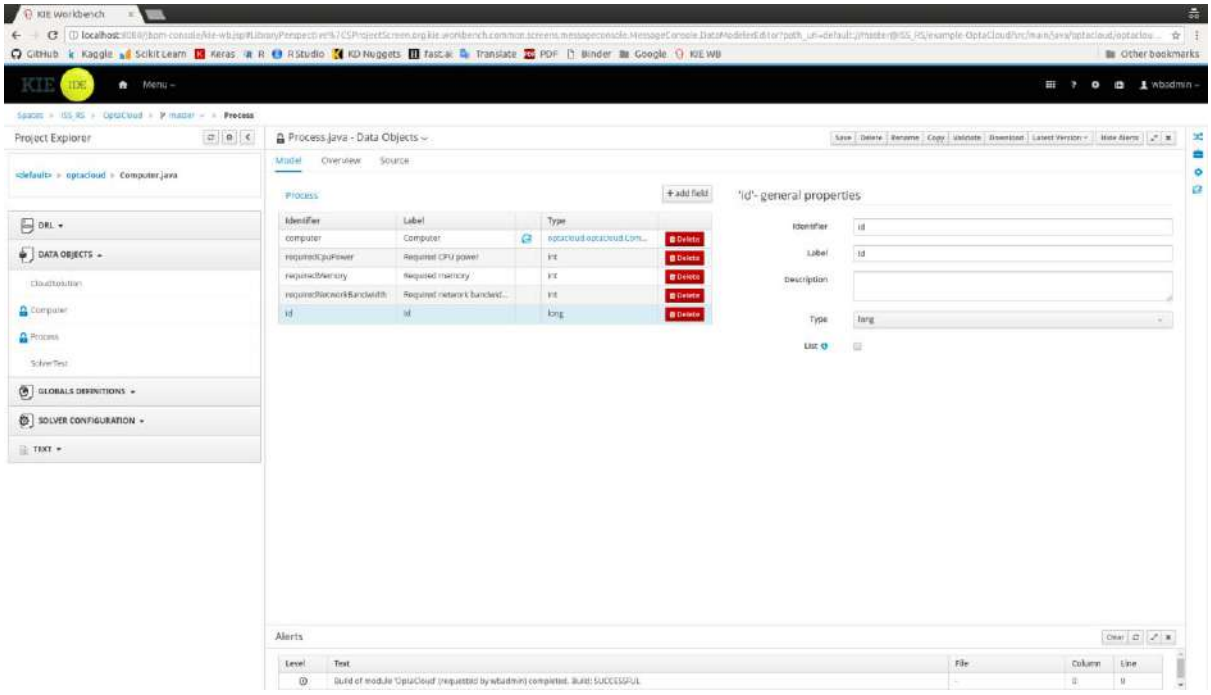


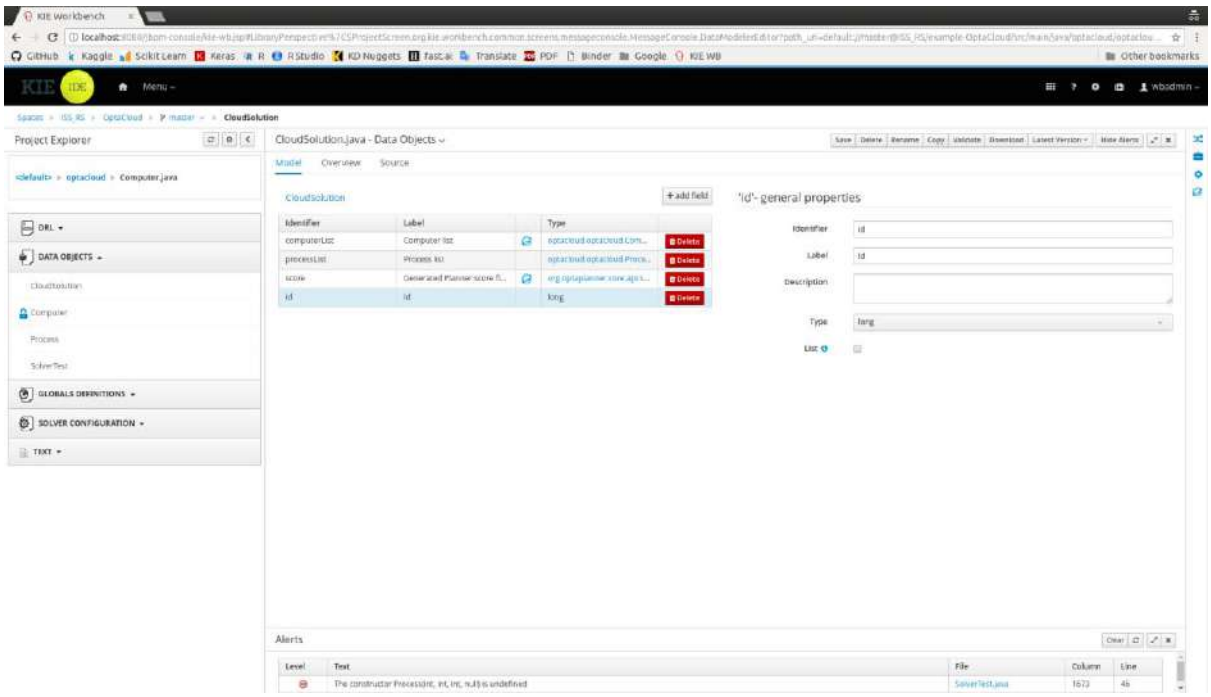
Above: try-sample project ***OptaCloud_ISS_RS*** in KIE Workbench

6) [Data Object] Add field: **id** (type: **long**) to Data Objects: **Computer**, **Process**, **CloudSolution**; Save all three objects;



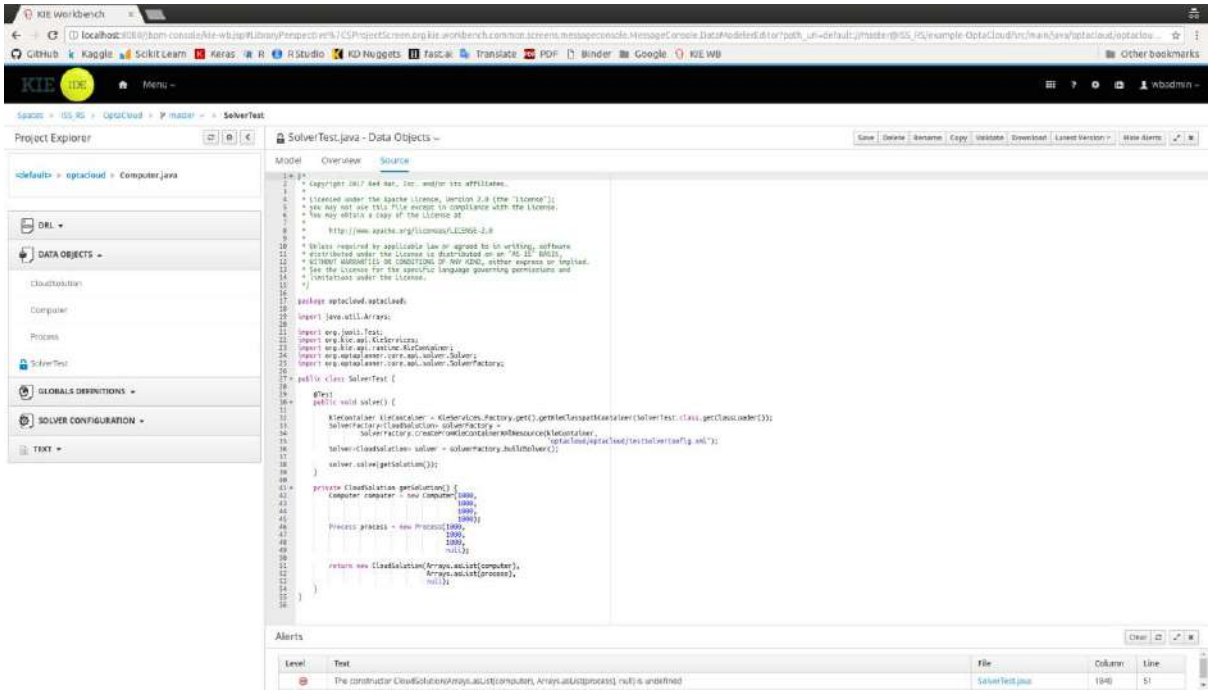




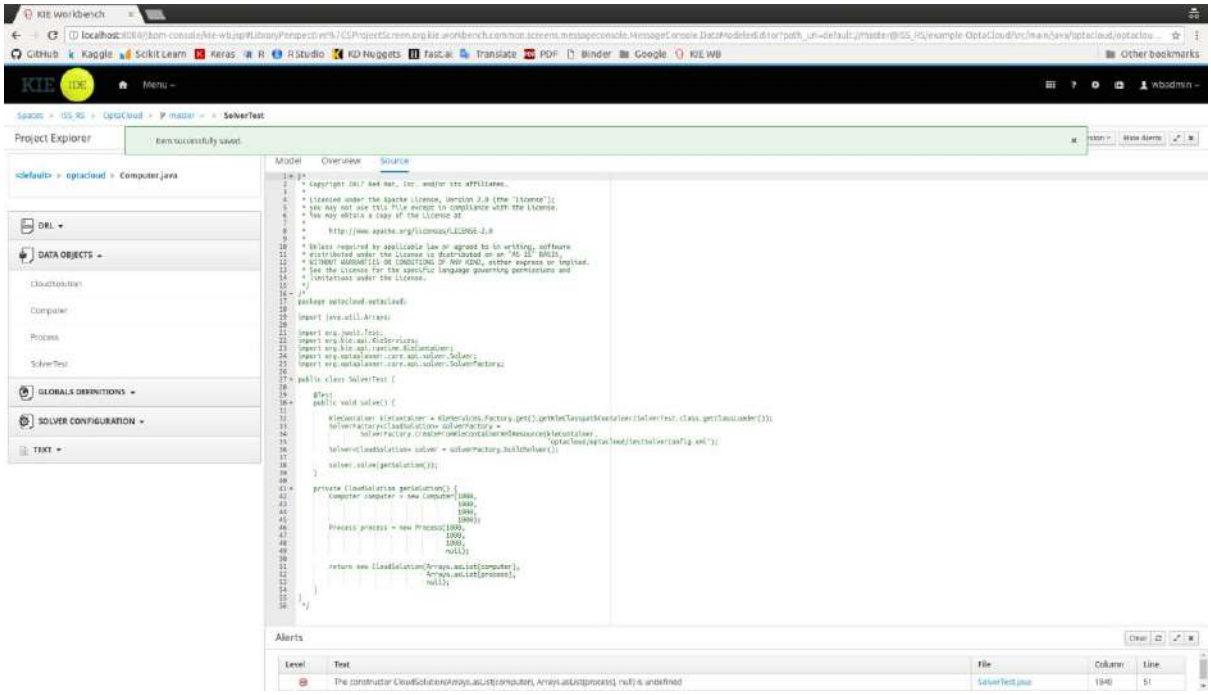


Above: domain objects based on class diagram

7) [Data Object] To prevent Build/Deploy error, comment all code in Java script: **example-OptaCloud/src/test/java/optacloud/optacloud/SolverTest.java**

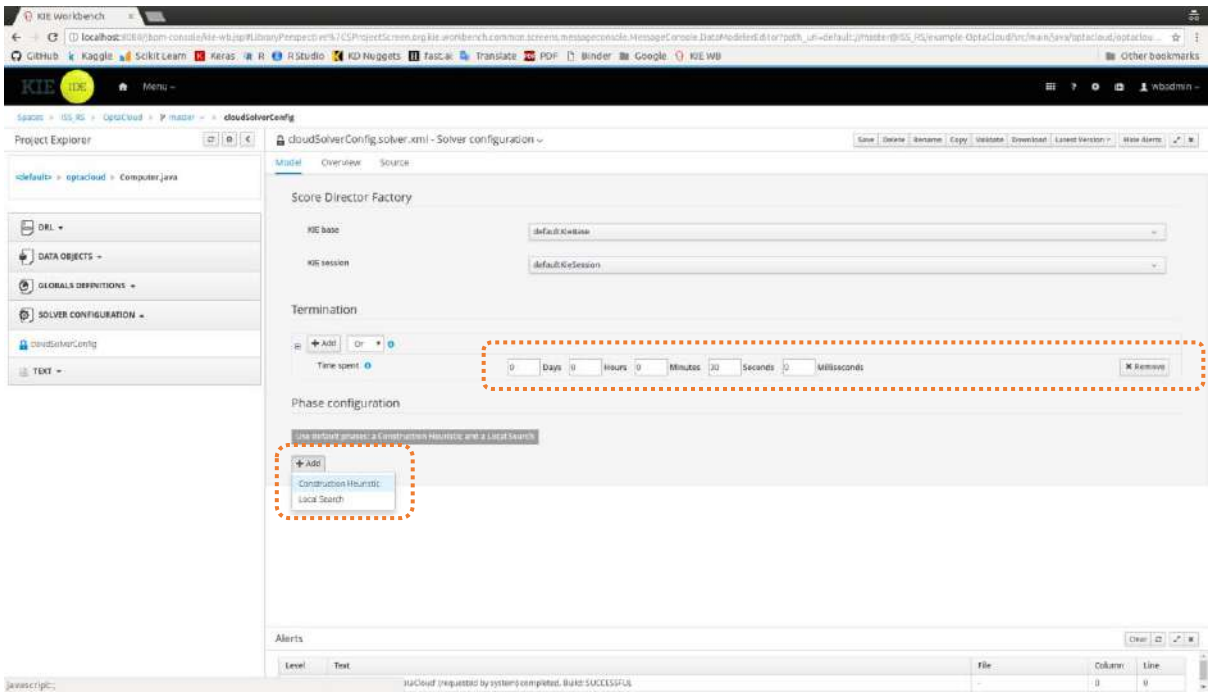


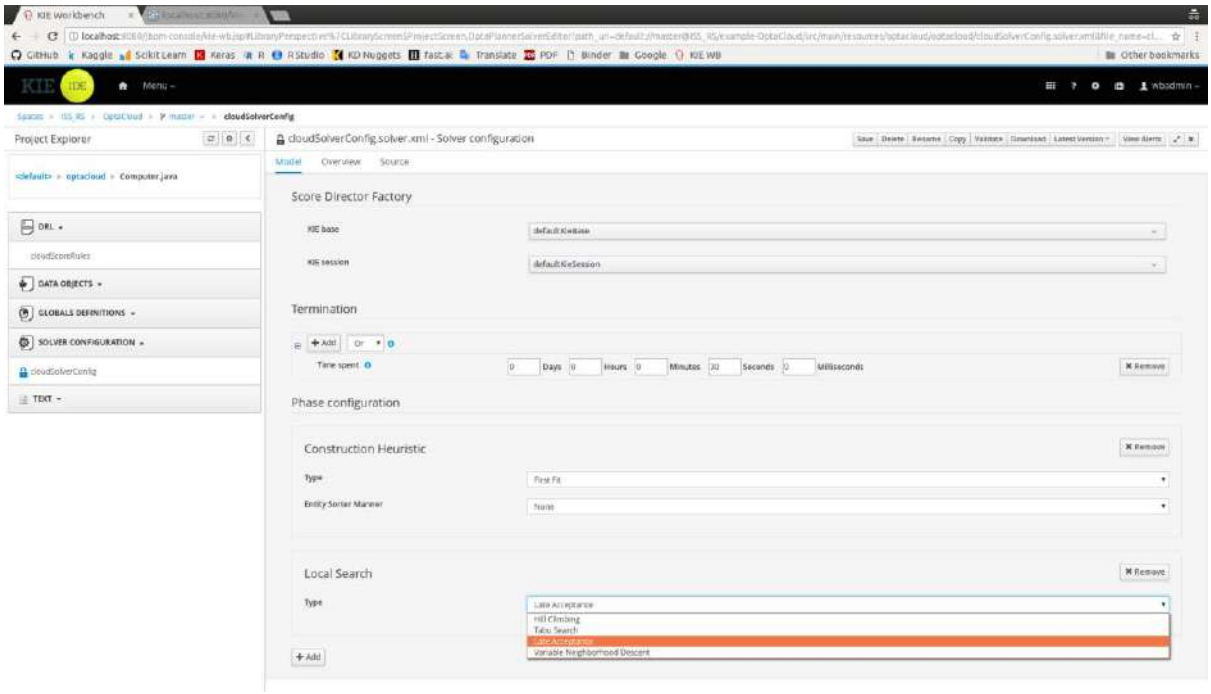
Save

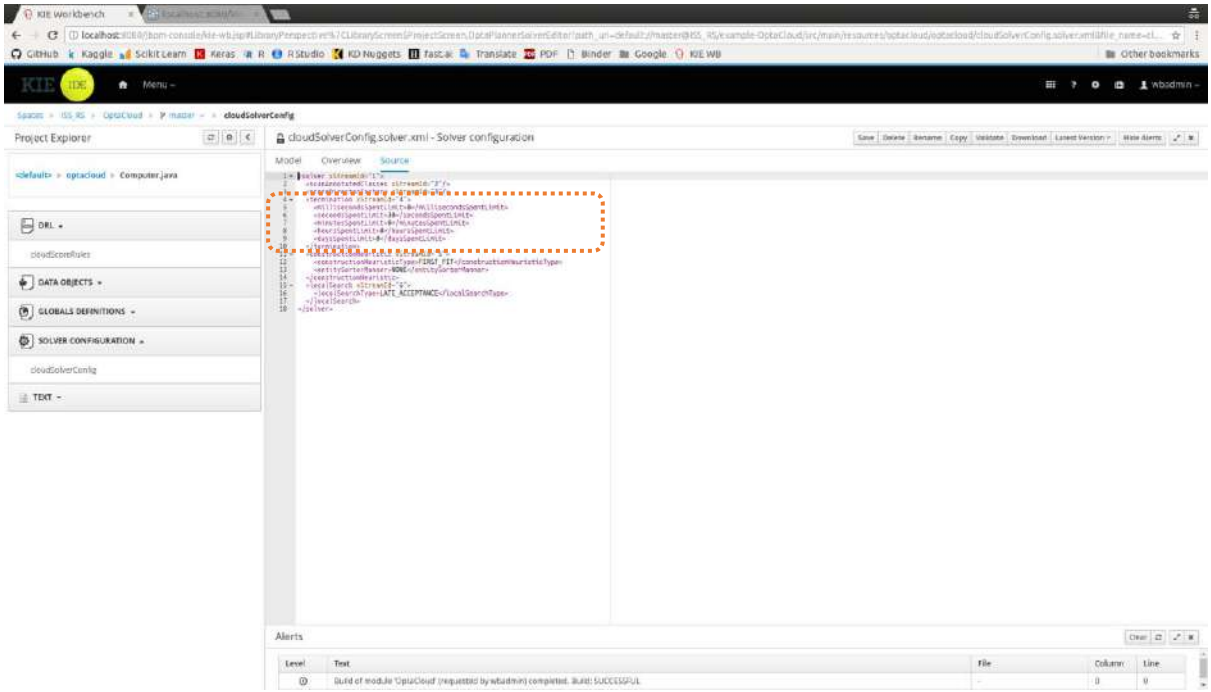


Above: unused ***SolverTest*** java program

8) [Solver Configuration] Update





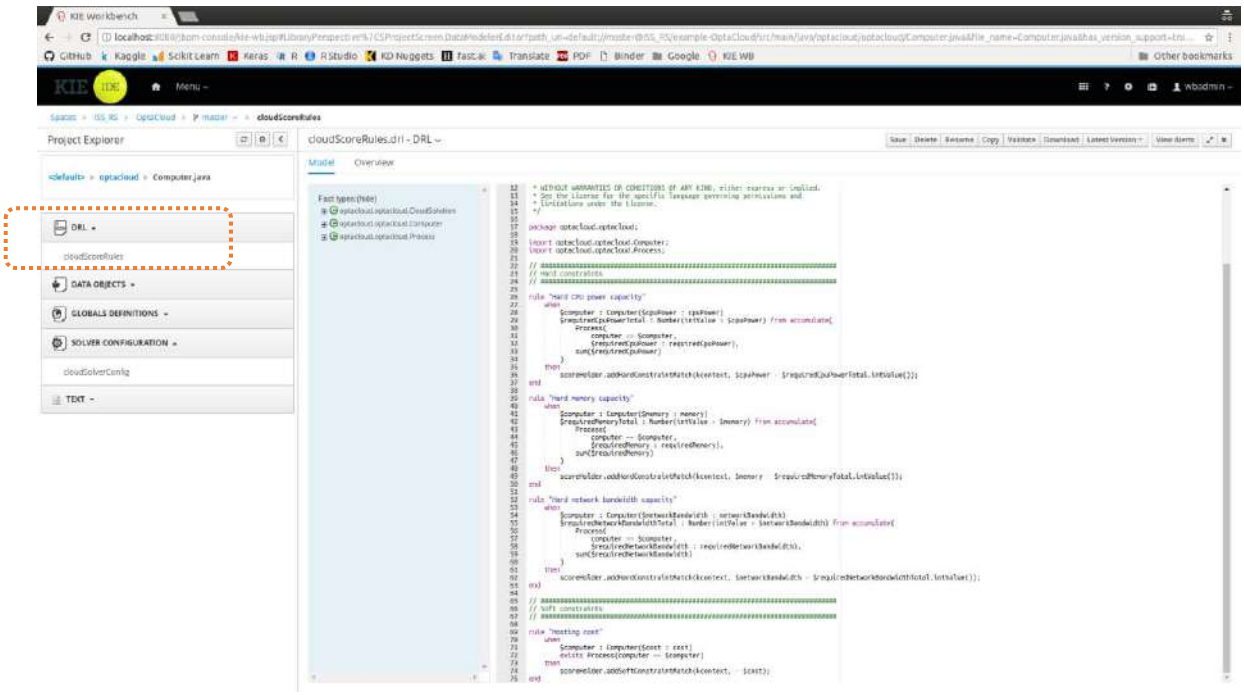


Above: solver configuration (xml)

example-OptaCloud/src/main/resources/optacloud/optacloud/cloudSolverConfig.solver.xml

```
<solver xStreamId="1">
  <scanAnnotatedClasses xStreamId="2"/>
  <scoreDirectorFactory xStreamId="3"/>
  <termination xStreamId="4">
    <millisecondsSpentLimit>0</millisecondsSpentLimit>
    <secondsSpentLimit>30</secondsSpentLimit>
    <minutesSpentLimit>0</minutesSpentLimit>
    <hoursSpentLimit>0</hoursSpentLimit>
    <daysSpentLimit>0</daysSpentLimit>
  </termination>
  <constructionHeuristic xStreamId="5">
    <constructionHeuristicType>FIRST_FIT</constructionHeuristicType>
    <entitySorterManner>NONE</entitySorterManner>
  </constructionHeuristic>
  <localSearch xStreamId="6">
    <localSearchType>LATE_ACCEPTANCE</localSearchType>
  </localSearch>
</solver>
```

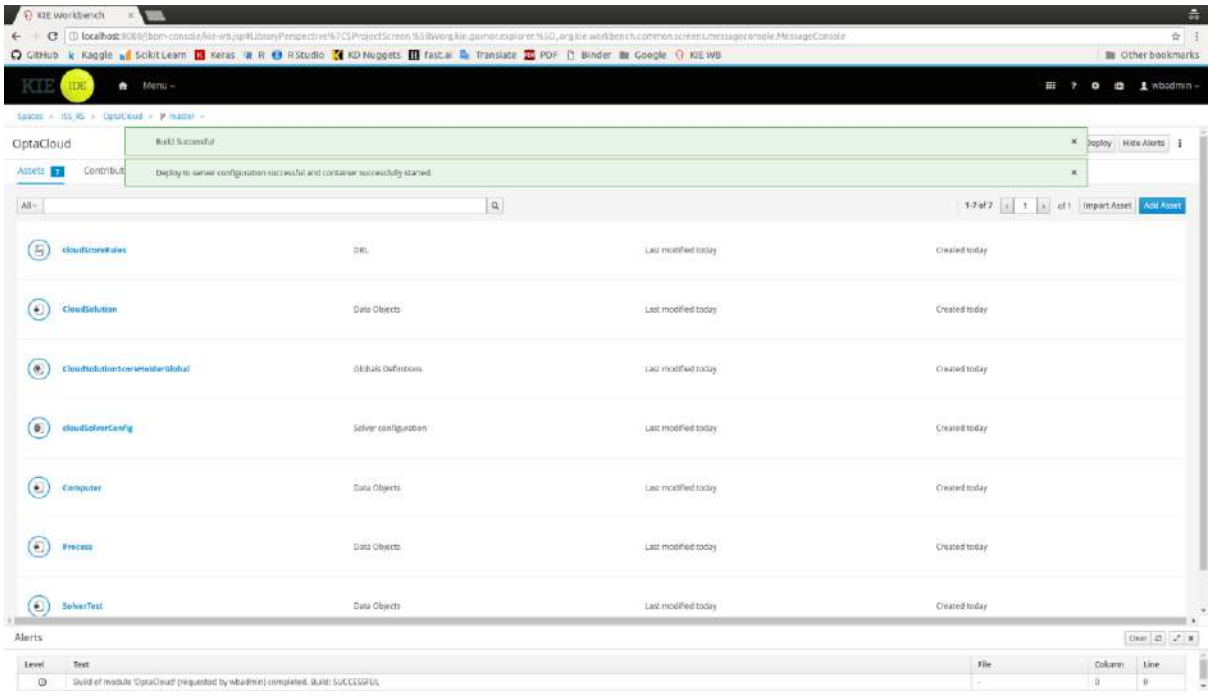
9) [DRL] Constraint Definition / Score Calculation using Drools Rule



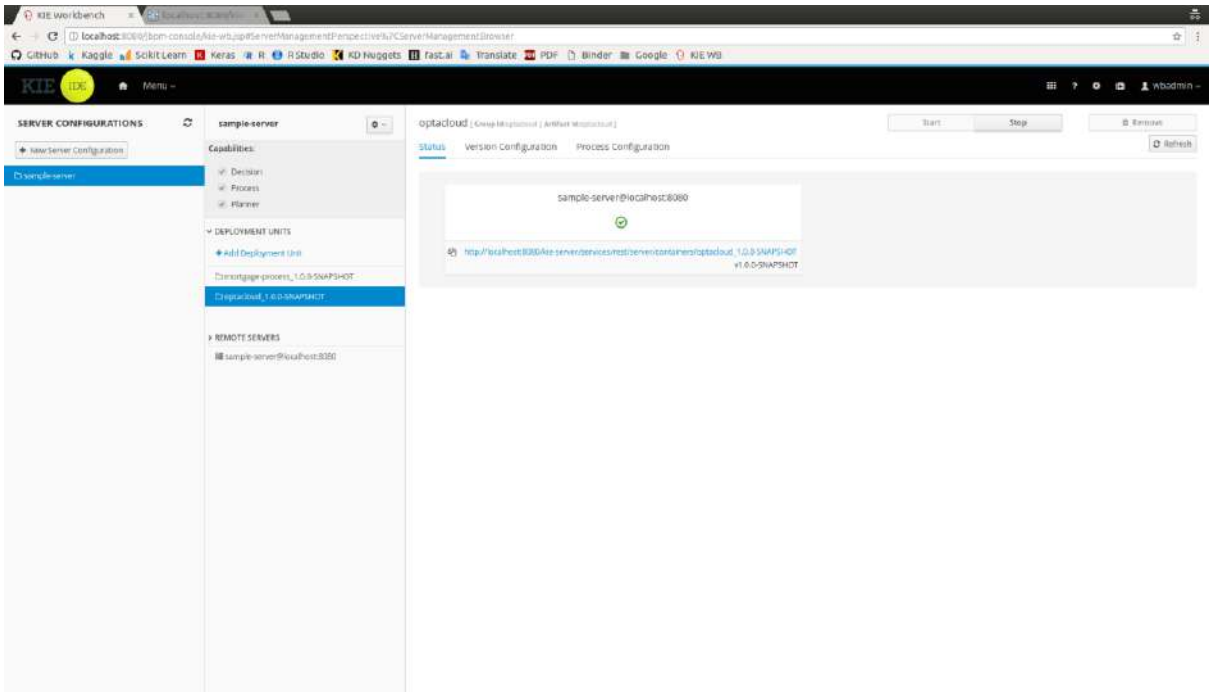
Above: constraints using Drools rule

10)[Deploy] Deploy OptaCloud solver to server

☺ Remember to **Save** all modified assets before build/deploy.



Deploy to KIE Server ***optacloud_1.0.0-SNAPSHOT***

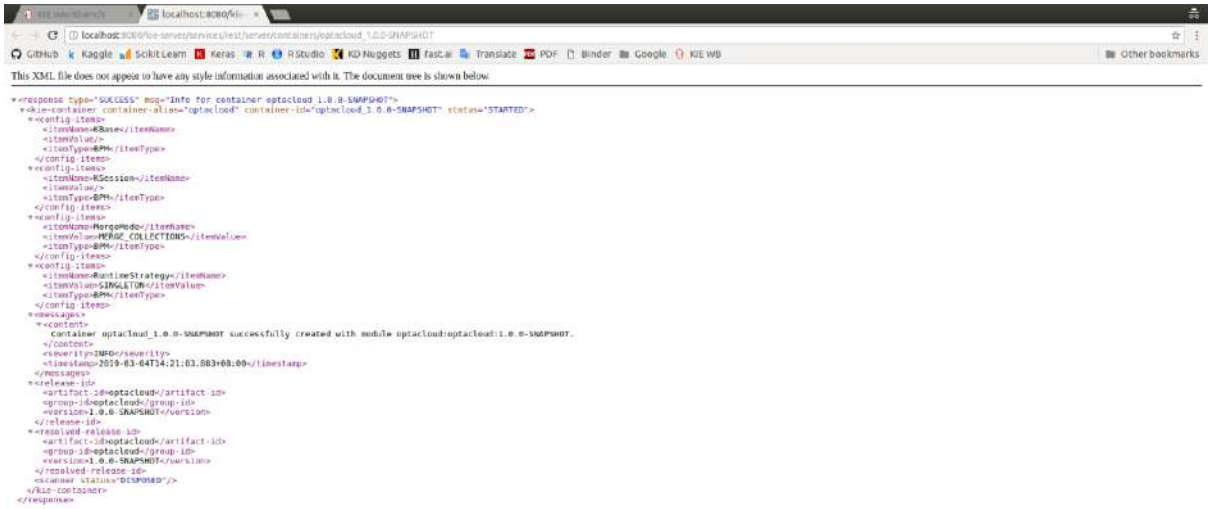


Above: solver deployment to KIE Server

http://localhost:8080/kie-server/services/rest/server/containers/optacloud_1.0.0-SNAPSHOT

Username: kieserver **Password:** kieserver1!

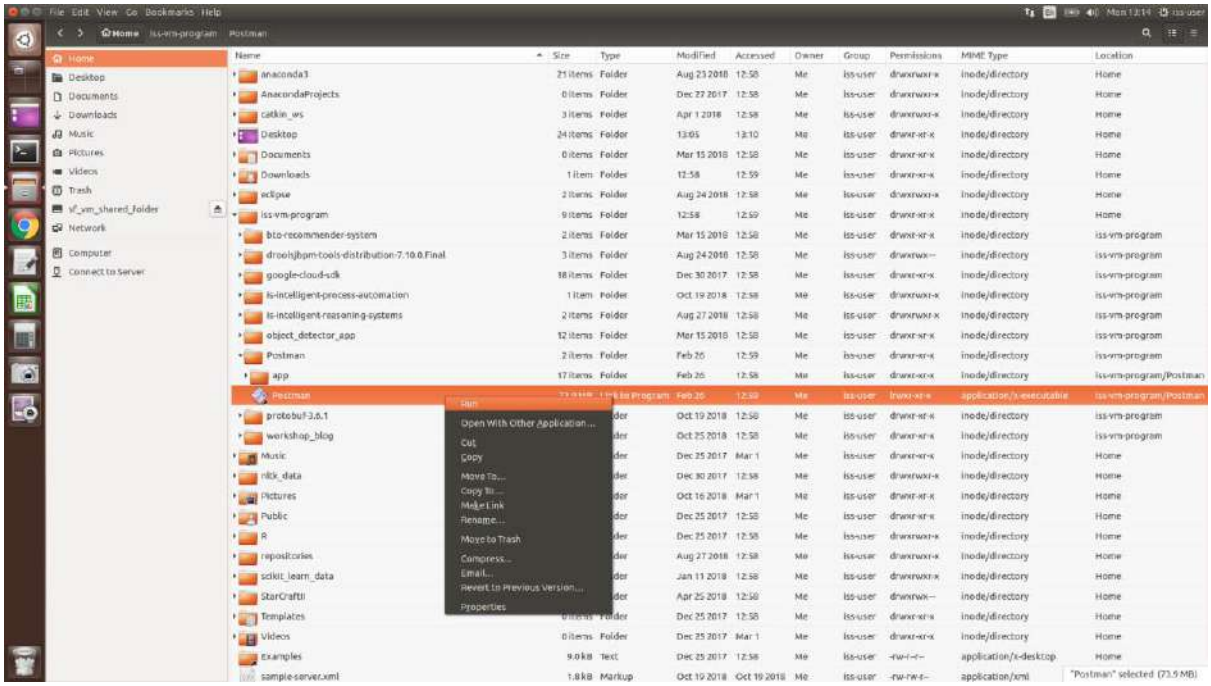
Above: solver web service end point

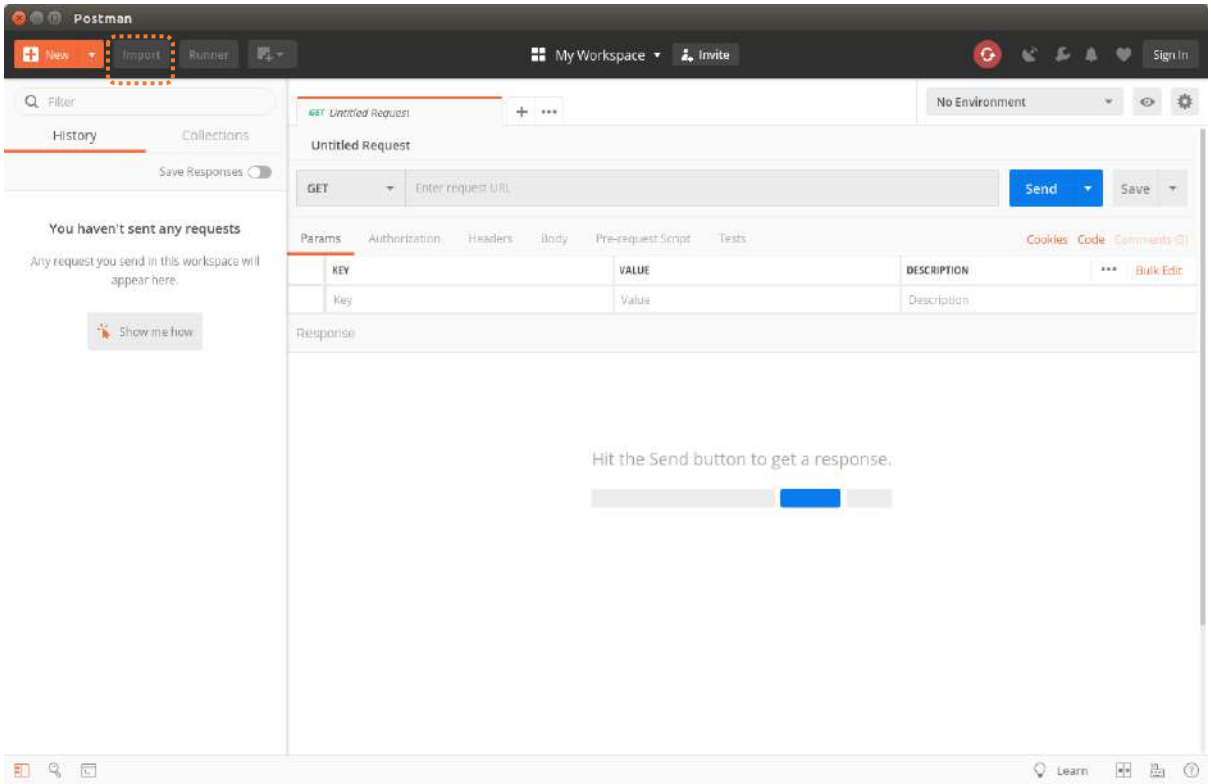


```
<response type="SUCCESS" msg="Info for container optacloud_1.0.0-SNAPSHOT">
<kie-container container-alias="optacloud" container-id="optacloud_1.0.0-
SNAPSHOT" status="STARTED">
<config-items>
<itemName>KBase</itemName>
<itemValue/>
<itemType>BPM</itemType>
</config-items>
<config-items>
<itemName>KSession</itemName>
<itemValue/>
<itemType>BPM</itemType>
</config-items>
<config-items>
<itemName>MergeMode</itemName>
<itemValue>MERGE_COLLECTIONS</itemValue>
<itemType>BPM</itemType>
</config-items>
<config-items>
<itemName>RuntimeStrategy</itemName>
<itemValue>SINGLETON</itemValue>
<itemType>BPM</itemType>
</config-items>
<messages>
<content>
Container optacloud_1.0.0-SNAPSHOT successfully created with module
optacloud:optacloud:1.0.0-SNAPSHOT.
</content>
<severity>INFO</severity>
<timestamp>2019-03-04T14:21:03.883+08:00</timestamp>
</messages>
<release-id>
<artifact-id>optacloud</artifact-id>
<group-id>optacloud</group-id>
<version>1.0.0-SNAPSHOT</version>
</release-id>
<resolved-release-id>
<artifact-id>optacloud</artifact-id>
<group-id>optacloud</group-id>
<version>1.0.0-SNAPSHOT</version>
</resolved-release-id>
<scanner status="DISPOSED"/>
</kie-container>
</response>
```

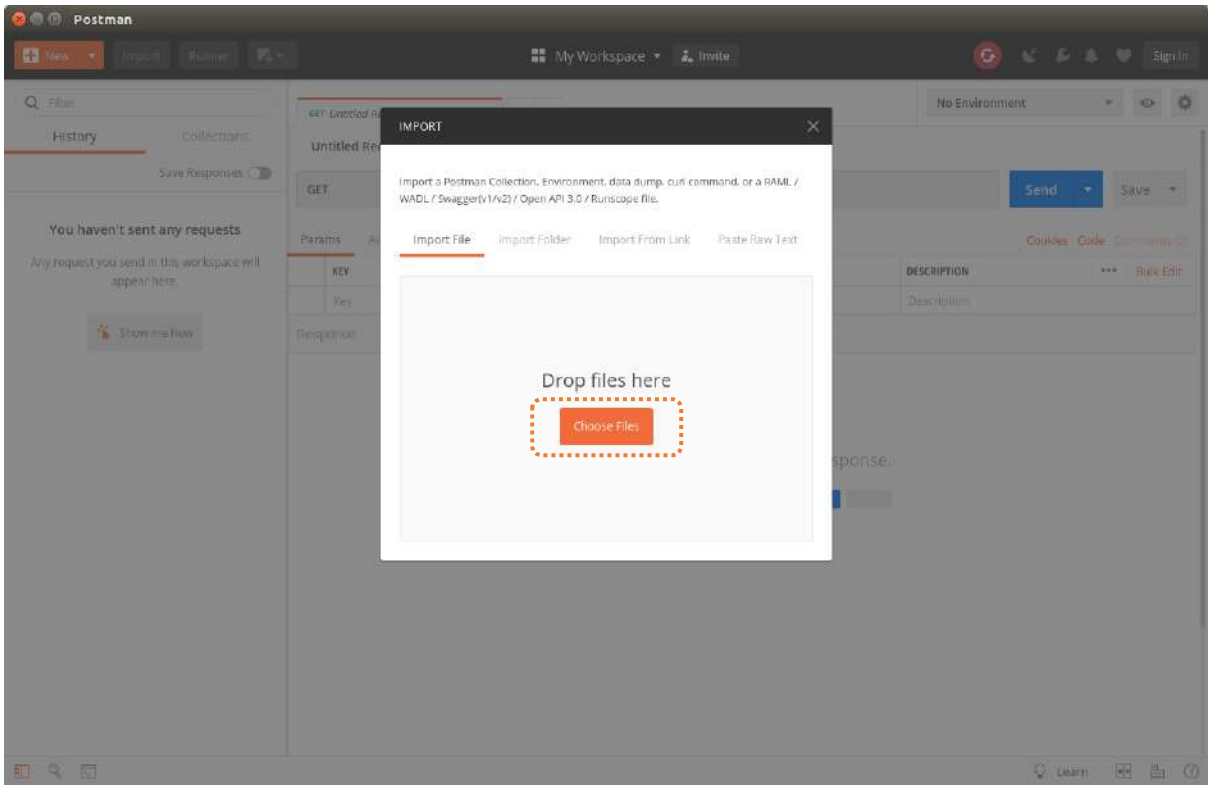
11)[API] Use RESTful API to interact with deployed OptaCloud solver

Start tool Postman

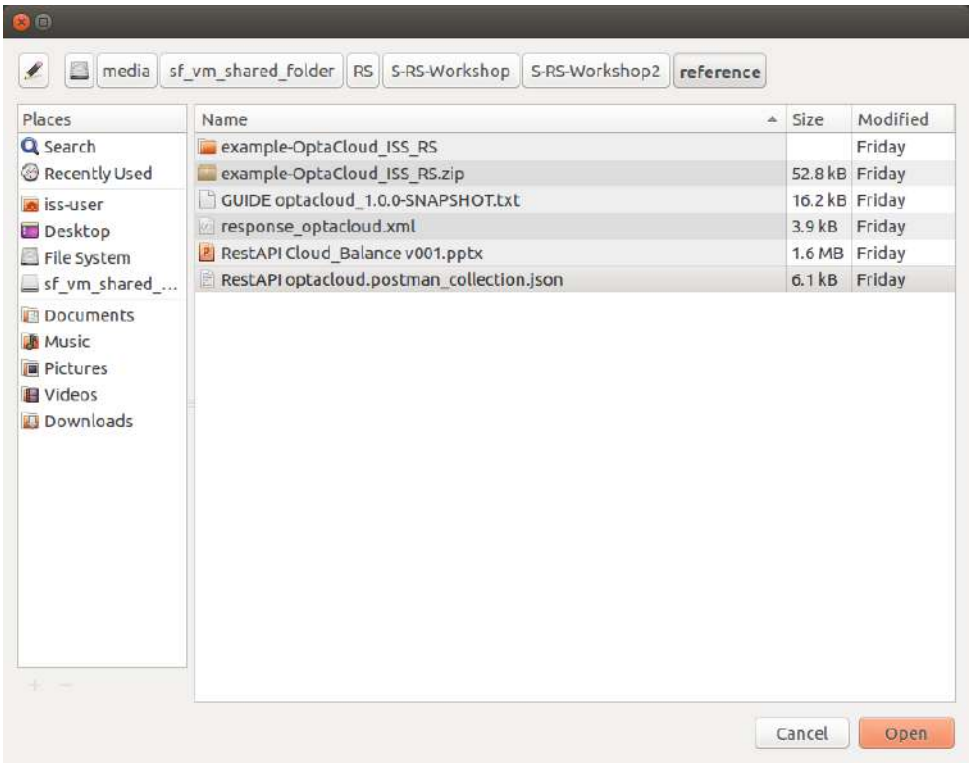


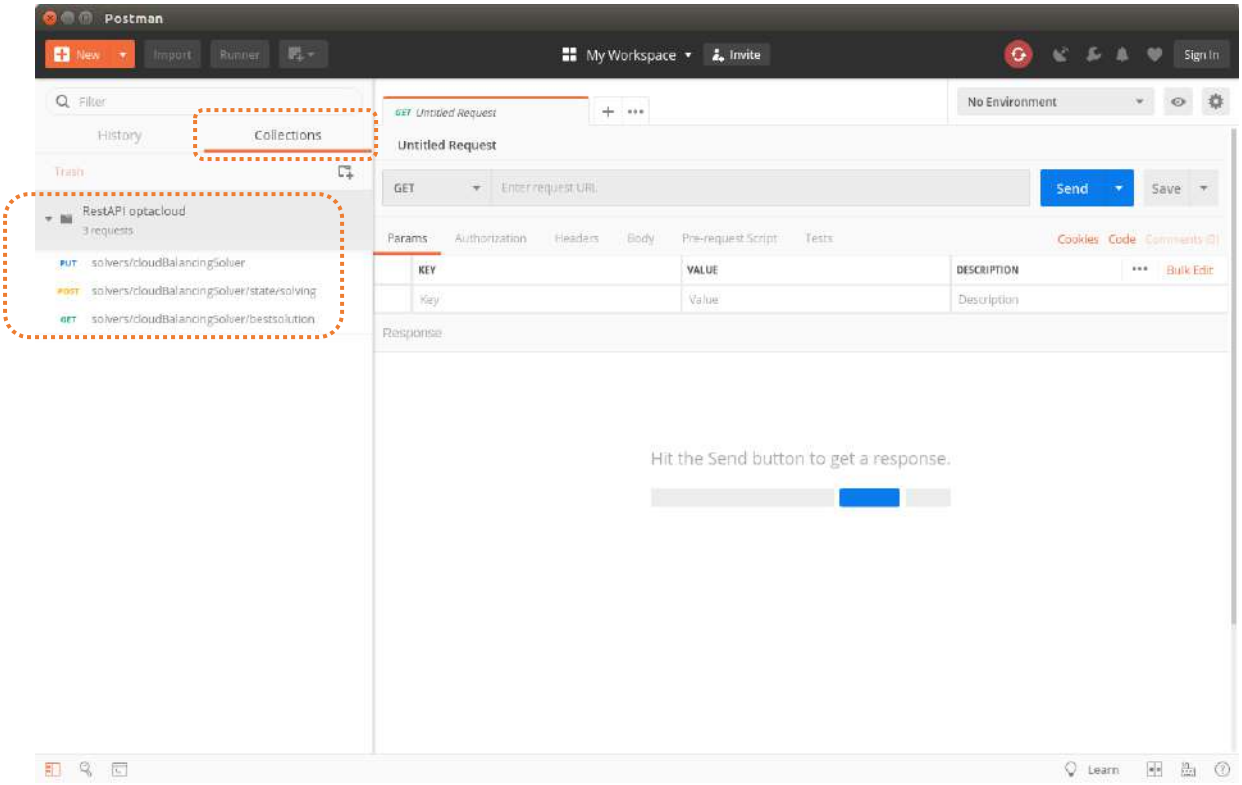


Import Pre-built API calls for OptaCloud project



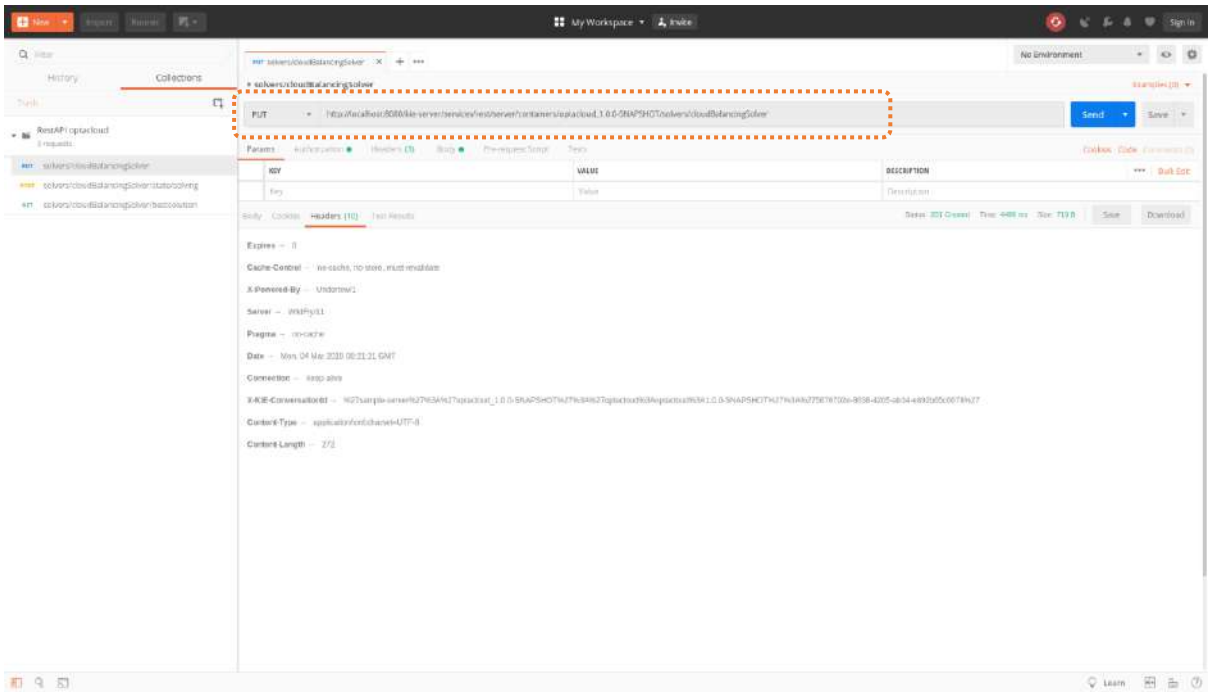
Select file: ***RestAPI optacloud v?.0.0.postman_collection.json***



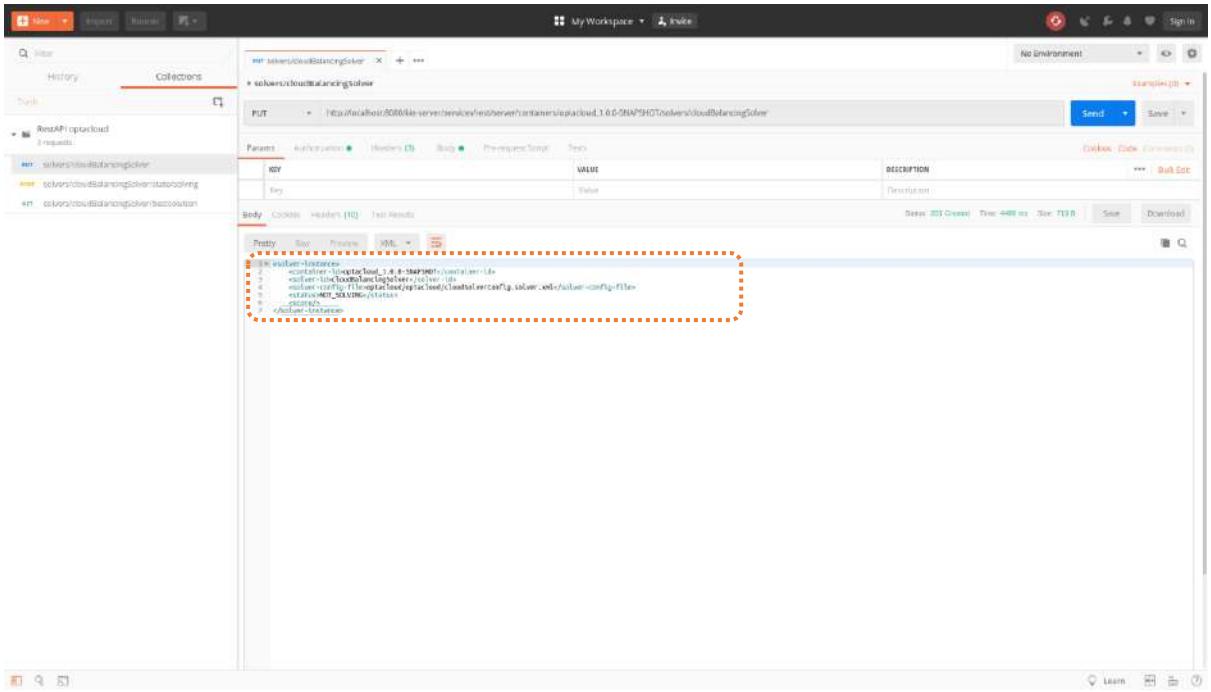


Above: Imported **PUT/POST/GET** APIs

PUT – Initiate a solver object/instance

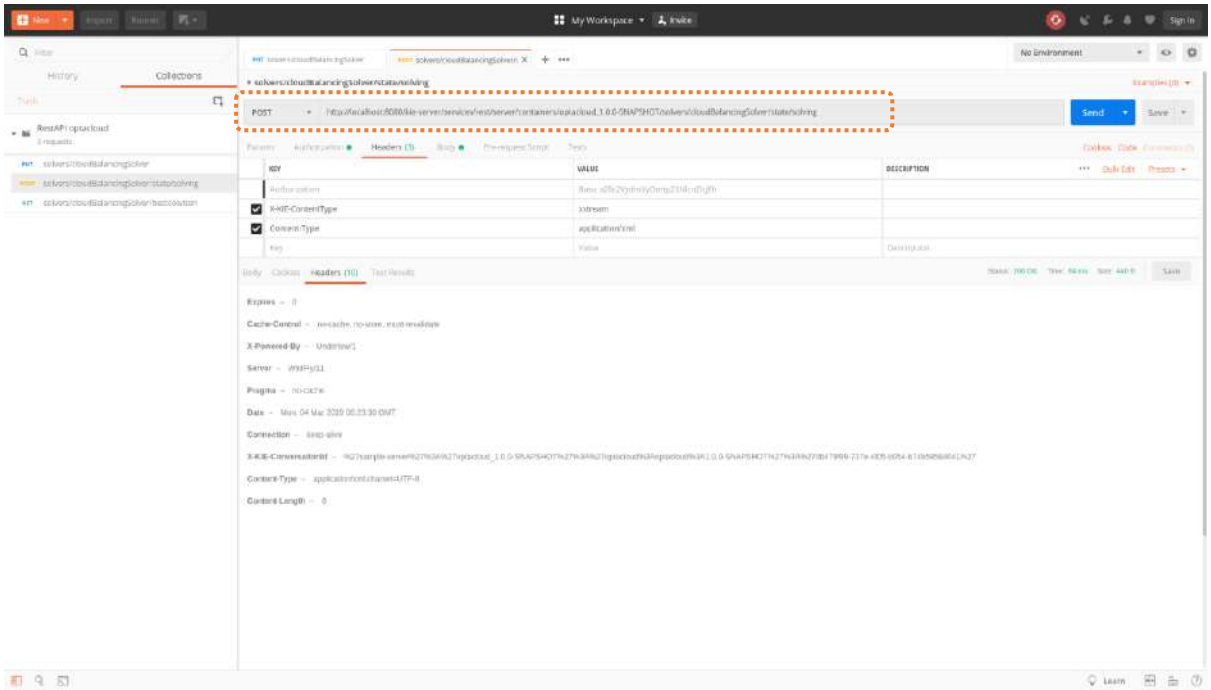


Above: REST-API PUT **Solver_Config** via Postman tool

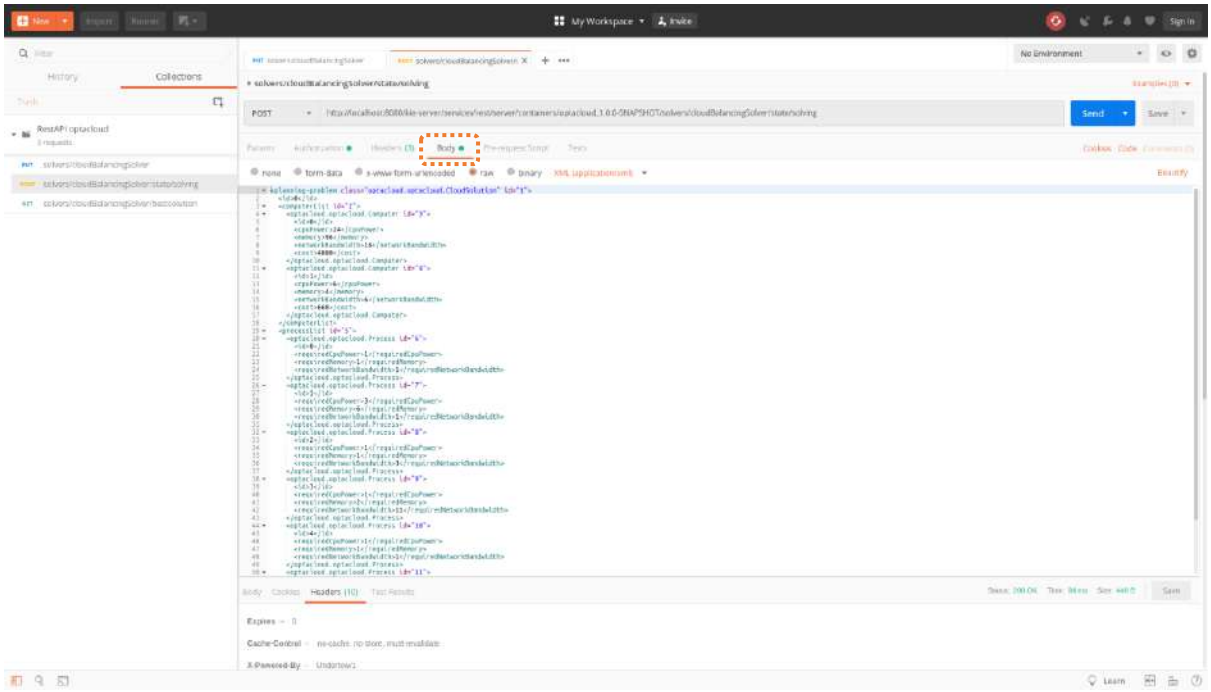


Above: REST-API PUT **Solver_Config** via Postman tool

```
<solver-instance>
  <solver-config-file>optacloud/optacloud/cloudSolverConfig.solver.xml</solver-config-file>
</solver-instance>
```



Above: REST-API POST **biz context** via Postman tool

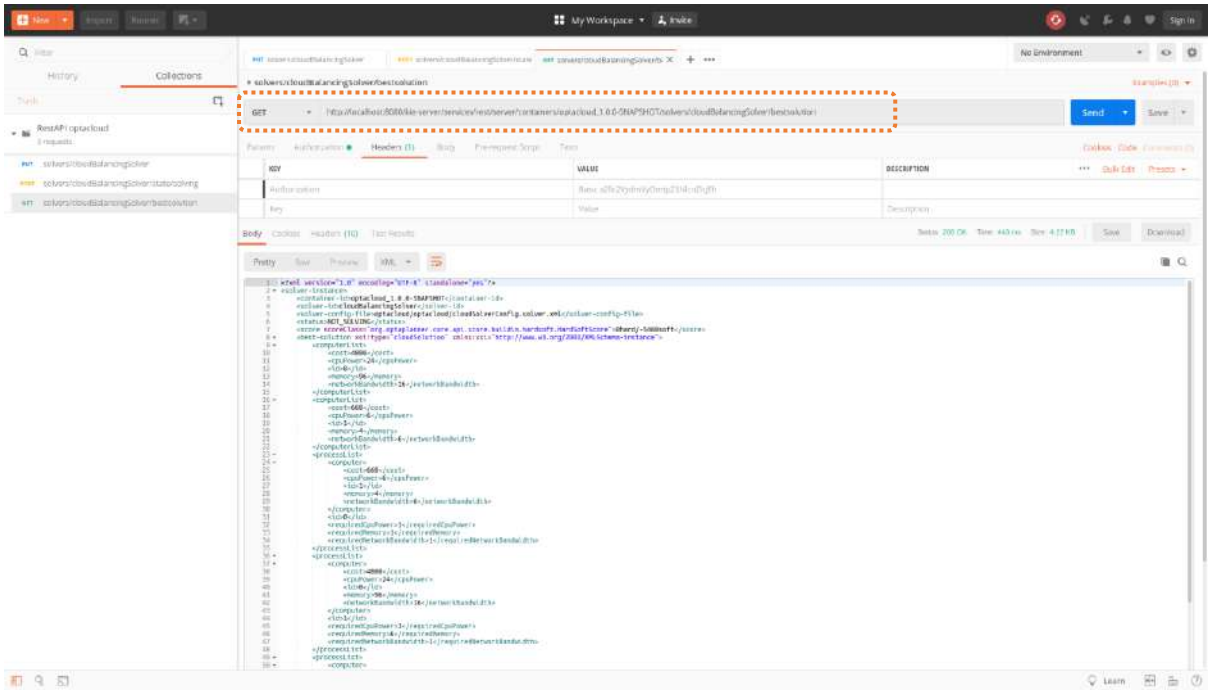


Above: REST-API POST **biz context** via Postman tool

```
<planning-problem class="optacloud.optacloud.CloudSolution" id="1">
  <id>0</id>
  <computerList id="2">
    <optacloud.optacloud.Computer id="3">
      <id>0</id>
      <cpuPower>24</cpuPower>
      <memory>96</memory>
      <networkBandwidth>16</networkBandwidth>
      <cost>4800</cost>
    </optacloud.optacloud.Computer>
    <optacloud.optacloud.Computer id="4">
      <id>1</id>
      <cpuPower>6</cpuPower>
      <memory>4</memory>
      <networkBandwidth>6</networkBandwidth>
      <cost>660</cost>
    </optacloud.optacloud.Computer>
  </computerList>
  <processList id="5">
    <optacloud.optacloud.Process id="6">
      <id>0</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="7">
      <id>1</id>
      <requiredCpuPower>3</requiredCpuPower>
      <requiredMemory>6</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="8">
      <id>2</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>3</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="9">
```

```
      <id>3</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>2</requiredMemory>
      <requiredNetworkBandwidth>11</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="10">
      <id>4</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="11">
      <id>5</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>5</requiredNetworkBandwidth>
    </optacloud.optacloud.Process>
  </processList>
</planning-problem>
```

GET – Obtain best solution (xml)



Above: REST-API GET **best solution** via Postman tool

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<solver-instance>
    <container-id>optacloud_1.0.0-SNAPSHOT</container-id>
    <solver-id>cloudBalancingSolver</solver-id>
    <solver-config-file>optacloud/optacloud/cloudSolverConfig.solver.xml</solver-config-file>
    <status>NOT_SOLVING</status>
    <score  scoreClass="org.optaplanner.core.api.score.buildin.hardsoft.HardSoftScore">0hard/-5460soft</score>
    <best-solution    xsi:type="cloudSolution"    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <computerList>
            <cost>4800</cost>
            <cpuPower>24</cpuPower>
            <id>0</id>
            <memory>96</memory>
            <networkBandwidth>16</networkBandwidth>
        </computerList>
        <computerList>
            <cost>660</cost>
            <cpuPower>6</cpuPower>
            <id>1</id>
            <memory>4</memory>
            <networkBandwidth>6</networkBandwidth>
        </computerList>
        <processList>
            <computer>
                <cost>660</cost>
                <cpuPower>6</cpuPower>
                <id>1</id>
                <memory>4</memory>
                <networkBandwidth>6</networkBandwidth>
            </computer>
            <id>0</id>
            <requiredCpuPower>1</requiredCpuPower>
            <requiredMemory>1</requiredMemory>
            <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
        </processList>
        <processList>
            <computer>
                <cost>4800</cost>
```

```
            <cpuPower>24</cpuPower>
            <id>0</id>
            <memory>96</memory>
            <networkBandwidth>16</networkBandwidth>
        </computer>
        <id>1</id>
        <requiredCpuPower>3</requiredCpuPower>
        <requiredMemory>6</requiredMemory>
        <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
    </processList>
    <processList>
        <computer>
            <cost>4800</cost>
            <cpuPower>24</cpuPower>
            <id>0</id>
            <memory>96</memory>
            <networkBandwidth>16</networkBandwidth>
        </computer>
        <id>2</id>
        <requiredCpuPower>1</requiredCpuPower>
        <requiredMemory>1</requiredMemory>
        <requiredNetworkBandwidth>3</requiredNetworkBandwidth>
    </processList>
    <processList>
        <computer>
            <cost>4800</cost>
            <cpuPower>24</cpuPower>
            <id>0</id>
            <memory>96</memory>
            <networkBandwidth>16</networkBandwidth>
        </computer>
        <id>3</id>
        <requiredCpuPower>1</requiredCpuPower>
        <requiredMemory>2</requiredMemory>
        <requiredNetworkBandwidth>11</requiredNetworkBandwidth>
    </processList>
    <processList>
        <computer>
            <cost>4800</cost>
```

```
                <cpuPower>24</cpuPower>
                <id>0</id>
                <memory>96</memory>
                <networkBandwidth>16</networkBandwidth>
            </computer>
            <id>4</id>
            <requiredCpuPower>1</requiredCpuPower>
            <requiredMemory>1</requiredMemory>
            <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
        </processList>
        <processList>
            <computer>
                <cost>660</cost>
                <cpuPower>6</cpuPower>
                <id>1</id>
                <memory>4</memory>
                <networkBandwidth>6</networkBandwidth>
            </computer>
            <id>5</id>
            <requiredCpuPower>1</requiredCpuPower>
            <requiredMemory>1</requiredMemory>
            <requiredNetworkBandwidth>5</requiredNetworkBandwidth>
        </processList>
        <score>0hard/-5460soft</score>
        <id>0</id>
    </best-solution>
</solver-instance>
```

Reference

- https://docs.optaplanner.org/7.12.0.Final/optaplanner-wb-es-docs/html_single/
- http://www.optaplanner.org/learn/useCases/cloudOptimization.html
- http://www.optaplanner.org/learn/slides/optaplanner-presentation/training.html#/4/26
- https://docs.jboss.org/optaplanner/release/latestFinal/optaplanner-wb-es-docs/html_single/

1.2. Cloud Balance Solver Enhancement

{ Objective } Enhance the cloud balance solution to address additional business considerations/constraints, e.g. deep learning process with GPU requirements; Data centre physical locations; Network latency, etc.

Reuse KIE Workbench OptaCloud REST-API project. Enrich the Cloud Balancing domain model and add extra constraints such as:

- Some Process running deep learning neural network models can require graphical processing units GPU chips, so these processes should (or must) be assigned to computers with sufficient number of GPU chips.
- Each Process belongs to a Service. A computer might crash, so processes running the same service must be assigned to different computers.
- Each Computer is located in a Building. A building might burn down, so processes of the same services should (or must) be assigned to computers in different buildings.

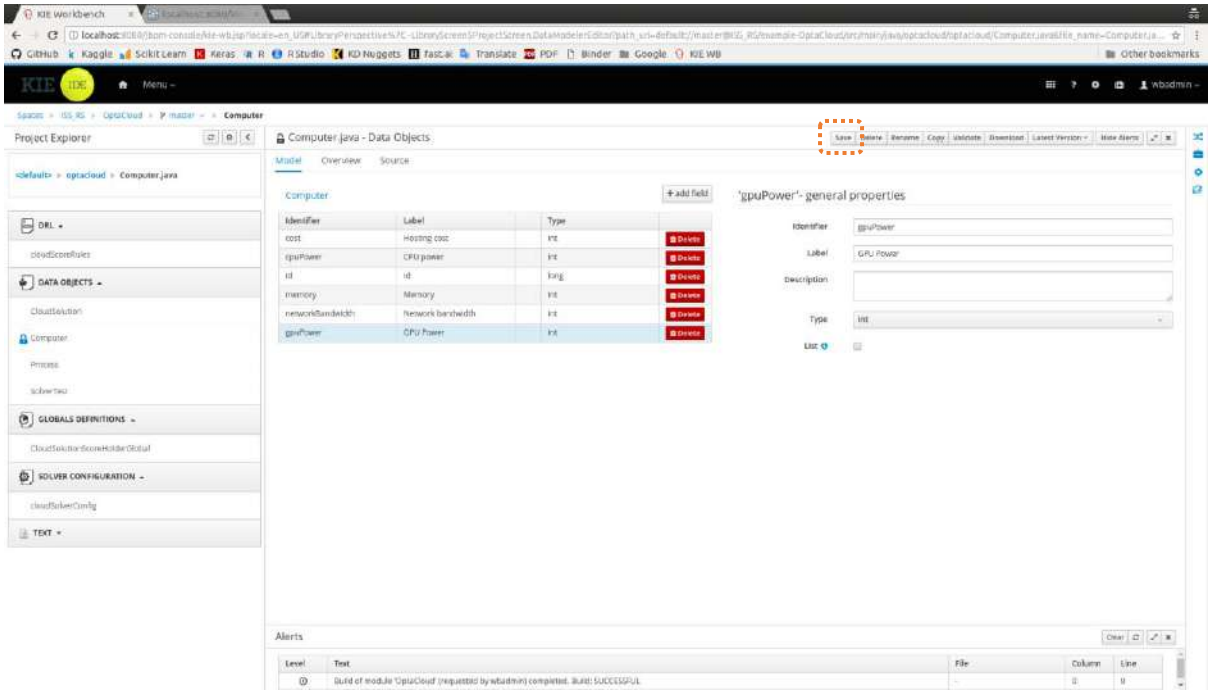
Reference

https://docs.optaplanner.org/latest/optaplanner-docs/html_single/index.html#cloudBalancingBeyondThisTutorial

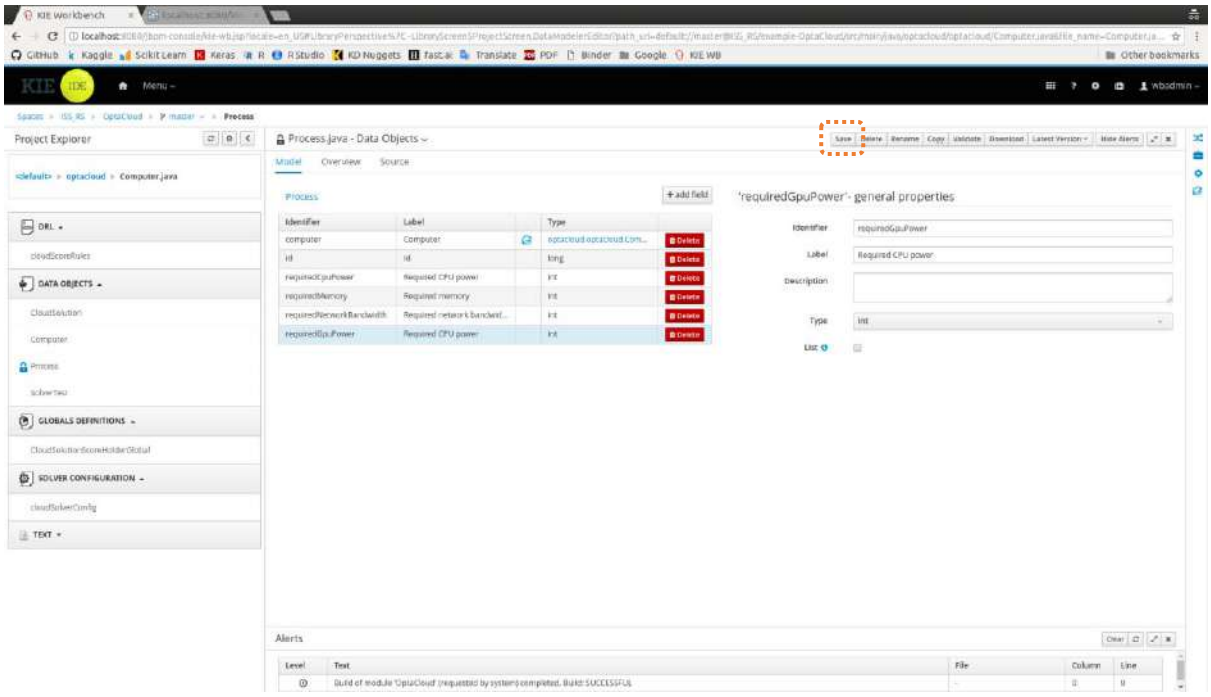
{ Reference Guide for GPU Enhancement }

1.2.1. Cloud Balance Solver Enhancement [Data Objects]

1) Add new ***gpuPower*** field for ***Computer*** data object; **Save**



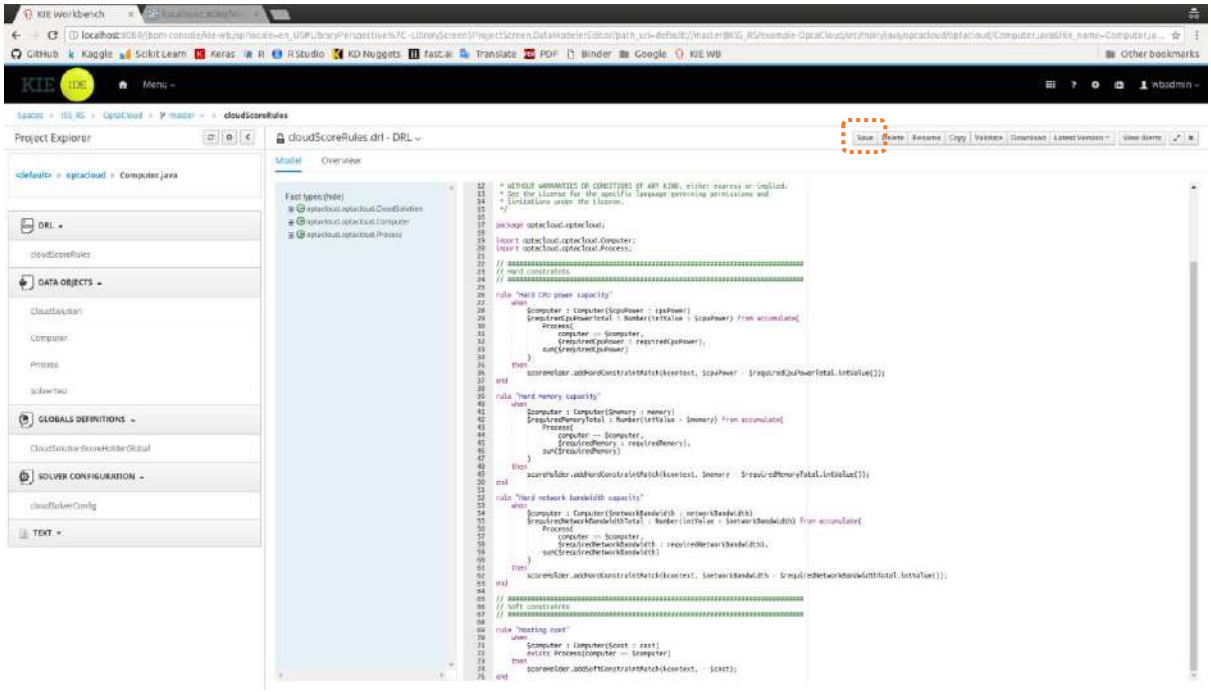
2) Add new ***requiredGpuPower*** field for ***Process*** data object; **Save**



1.2.2. Cloud Balance Solver Enhancement [DRL Rule]

1) Update DRL rule to include new constraint ***GPU*** into score calculation; **Save**

{ Quiz } Should the ***GPU*** be treated as **hard constraint** or **soft constraint**?



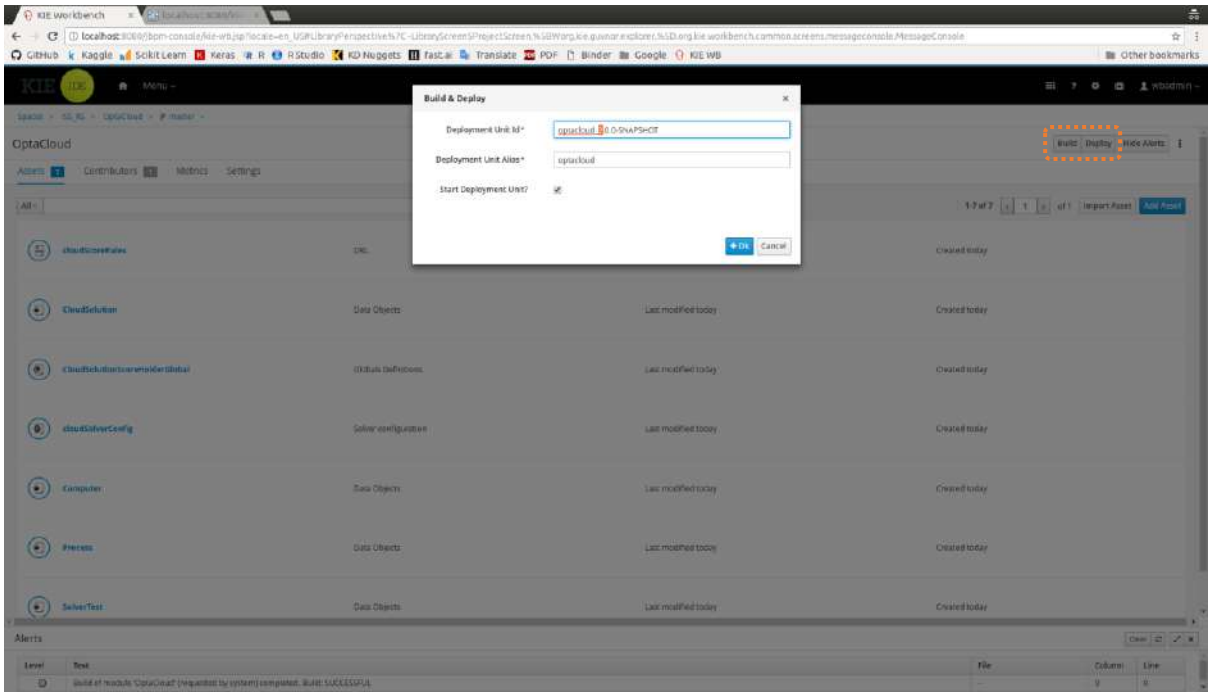
```
/*
 * Copyright 2017 Red Hat, Inc. and/or its affiliates.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package optacloud.optacloud;

import optacloud.optacloud.Computer;
import optacloud.optacloud.Process;

// ############################################################################
// Hard constraints
// ############################################################################

rule "Hard CPU power capacity"
    when
        $computer : Computer($cpuPower : cpuPower)
        $requiredCpuPowerTotal : Number(intValue > $cpuPower) from accumulate(
            Process(
                computer == $computer,
                $requiredCpuPower : requiredCpuPower),
            sum($requiredCpuPower)
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $cpuPower -
$requiredCpuPowerTotal.intValue());
end
```

```
// ------------------------------------------------------------------------
// 2019 March 04 - GU Zhan (Sam) - GPU constraints - START

rule "Hard GPU power capacity"
    when
        $computer : Computer($gpuPower : gpuPower)
        $requiredGpuPowerTotal : Number(intValue > $gpuPower) from accumulate(
            Process(
                computer == $computer,
                $requiredGpuPower : requiredGpuPower),
            sum($requiredGpuPower)
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $gpuPower -
$requiredGpuPowerTotal.intValue());
end

// 2019 March 04 - GU Zhan (Sam) - GPU constraints - E N D
// ------------------------------------------------------------------------

rule "Hard memory capacity"
    when
        $computer : Computer($memory : memory)
        $requiredMemoryTotal : Number(intValue > $memory) from accumulate(
            Process(
                computer == $computer,
                $requiredMemory : requiredMemory),
            sum($requiredMemory)
        )
    then
        scoreHolder.addHardConstraintMatch(kcontext, $memory -
$requiredMemoryTotal.intValue());
end

rule "Hard network bandwidth capacity"
    when
        $computer : Computer($networkBandwidth : networkBandwidth)
        $requiredNetworkBandwidthTotal : Number(intValue > $networkBandwidth) from accumulate(
            Process(
                computer == $computer,
```

```
                $requiredNetworkBandwidth : requiredNetworkBandwidth),

            sum($requiredNetworkBandwidth)

        )

    then

        scoreHolder.addHardConstraintMatch(kcontext, $networkBandwidth -
$requiredNetworkBandwidthTotal.intValue());

end


// ############################################################################

// Soft constraints

// ############################################################################


rule "Hosting cost"

    when

        $computer : Computer($cost : cost)

        exists Process(computer == $computer)

    then

        scoreHolder.addSoftConstraintMatch(kcontext, - $cost);

end
```

1.2.3. Cloud Balance Solver Enhancement [Deploy] v2.0.0

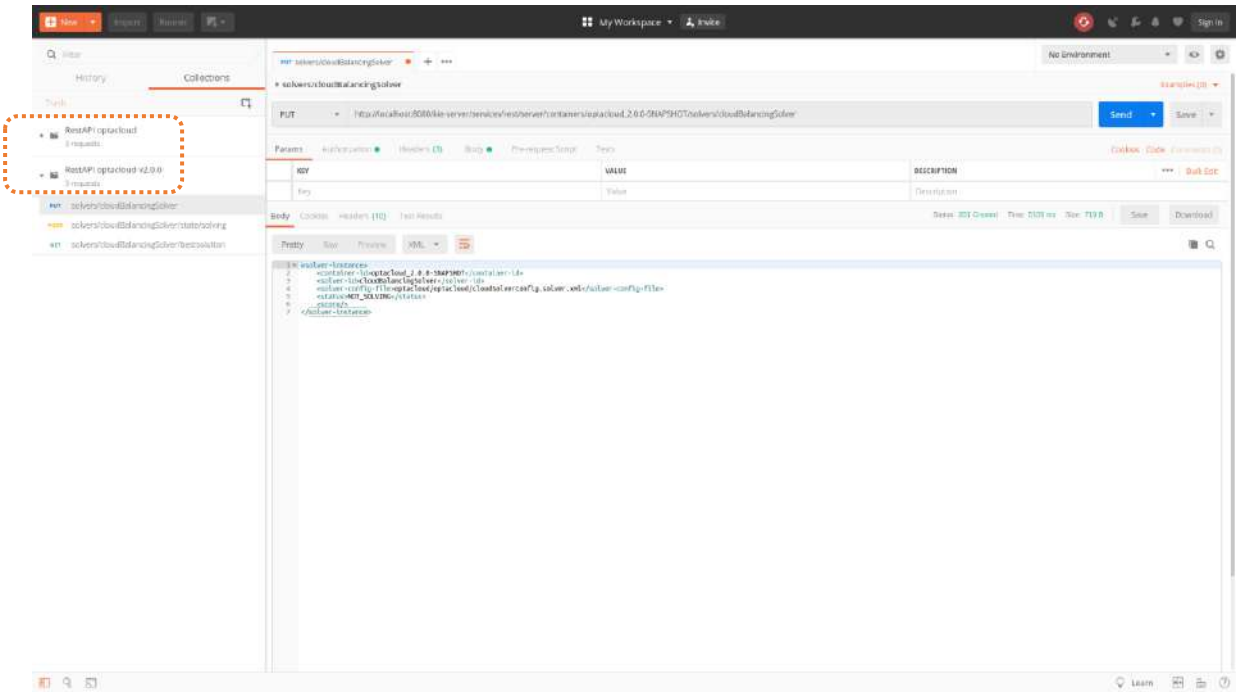
1) [Deploy] Deploy OptaCloud solver to server



Deploy to KIE Server ***optacloud_2.0.0-SNAPSHOT***

1.2.4. Cloud Balance Solver Enhancement [API]

1) [API: PUT] Initialize a new solver instance;



2) [API: POST] Provide a new business problem/context (with GPU requirements);

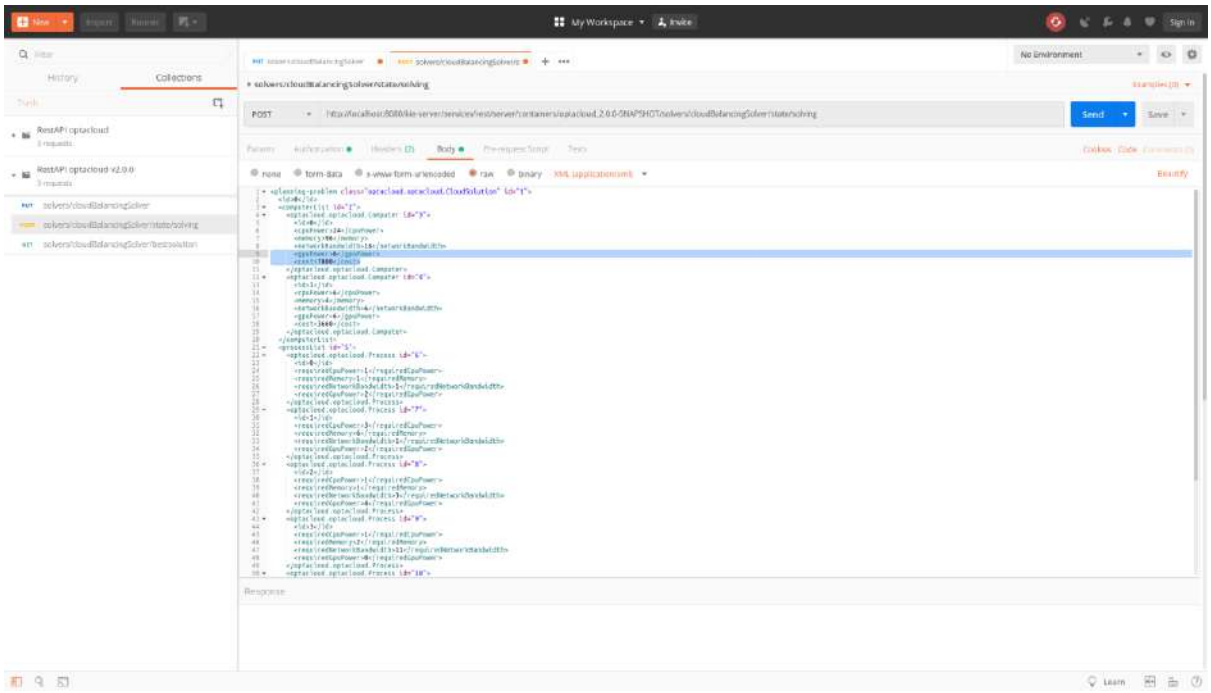
Update XML in API Body to introduce a new cloud balance problem:

Two computers with GPUs:

- Computer 0 has 6 GPU cards.
- Computer 1 has 6 GPU cards.

Six processes to assign to computer:

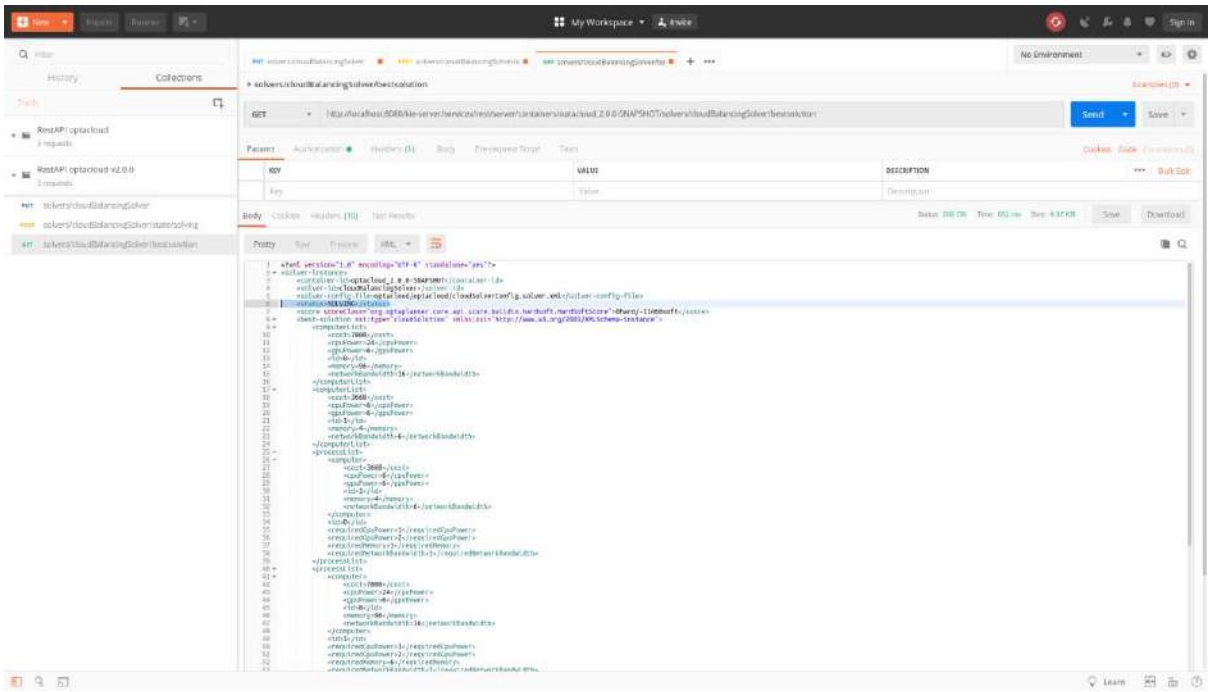
- Process 0 requires 2 GPU cards.
- Process 1 requires 2 GPU cards.
- Process 2 requires 4 GPU cards.
- Process 3 requires 0 GPU cards.
- Process 4 requires 0 GPU cards.
- Process 5 requires 0 GPU cards.



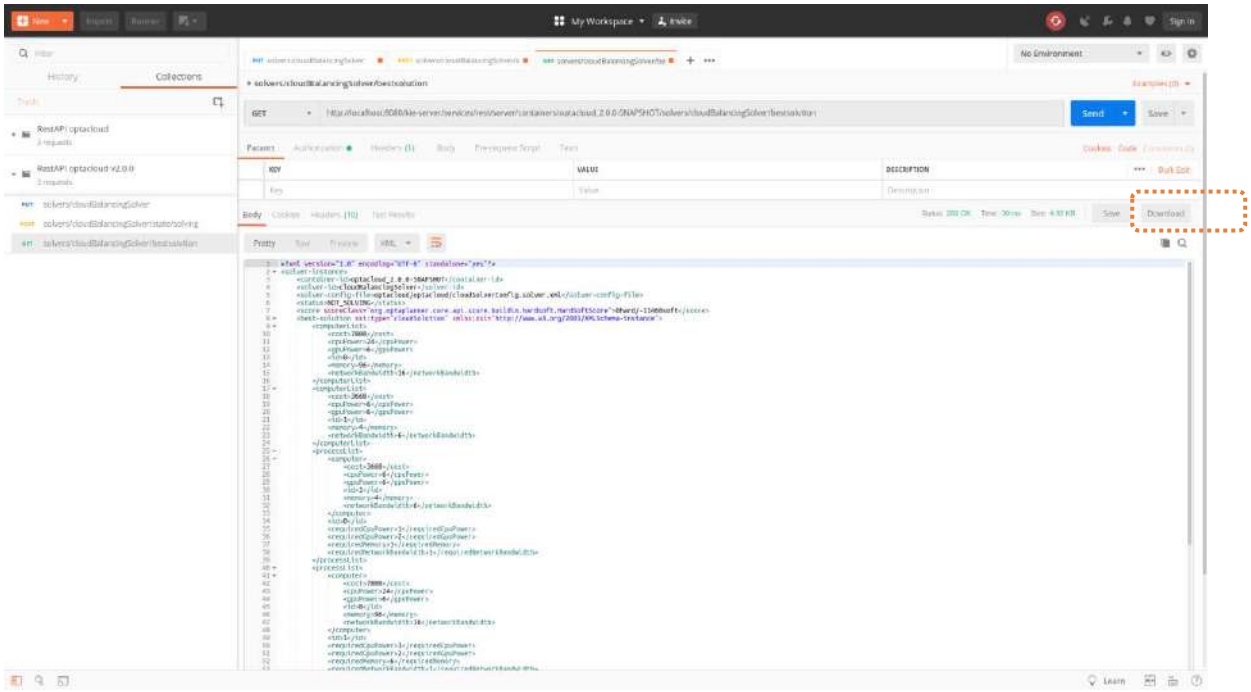
```
<planning-problem class="optacloud.optacloud.CloudSolution" id="1">
  <id>0</id>
  <computerList id="2">
    <optacloud.optacloud.Computer id="3">
      <id>0</id>
      <cpuPower>24</cpuPower>
      <memory>96</memory>
      <networkBandwidth>16</networkBandwidth>
      <gpuPower>6</gpuPower>
      <cost>7800</cost>
    </optacloud.optacloud.Computer>
    <optacloud.optacloud.Computer id="4">
      <id>1</id>
      <cpuPower>6</cpuPower>
      <memory>4</memory>
      <networkBandwidth>6</networkBandwidth>
      <gpuPower>6</gpuPower>
      <cost>3660</cost>
    </optacloud.optacloud.Computer>
  </computerList>
  <processList id="5">
    <optacloud.optacloud.Process id="6">
      <id>0</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
      <requiredGpuPower>2</requiredGpuPower>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="7">
      <id>1</id>
      <requiredCpuPower>3</requiredCpuPower>
      <requiredMemory>6</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
      <requiredGpuPower>2</requiredGpuPower>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="8">
      <id>2</id>
      <requiredCpuPower>1</requiredCpuPower>
```

```
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>3</requiredNetworkBandwidth>
      <requiredGpuPower>4</requiredGpuPower>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="9">
      <id>3</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>2</requiredMemory>
      <requiredNetworkBandwidth>11</requiredNetworkBandwidth>
      <requiredGpuPower>0</requiredGpuPower>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="10">
      <id>4</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>1</requiredNetworkBandwidth>
      <requiredGpuPower>0</requiredGpuPower>
    </optacloud.optacloud.Process>
    <optacloud.optacloud.Process id="11">
      <id>5</id>
      <requiredCpuPower>1</requiredCpuPower>
      <requiredMemory>1</requiredMemory>
      <requiredNetworkBandwidth>5</requiredNetworkBandwidth>
      <requiredGpuPower>0</requiredGpuPower>
    </optacloud.optacloud.Process>
  </processList>
</planning-problem>
```

3) [API: GET] Retrieve best solution (with GPU requirements);



<status>SOLVING</status>



<status>NOT_SOLVING</status>

{ Quiz } What controls solver to stop?

[Optional] { Challenge } Update the last three process to require 2 GPUs respectively; Run the solver to get solution; Observe the final score: Are all constraints (business resource requirements) satisfied?

Update XML in API Body to introduce a new cloud balance problem:

Two computers with GPUs:

- Computer 0 has 6 GPU cards.
- Computer 1 has 6 GPU cards.

Six processes to assign to computer:

- Process 0 requires 2 GPU cards.
- Process 1 requires 2 GPU cards.
- Process 2 requires 4 GPU cards.
- Process 3 requires 2 GPU cards.
- Process 4 requires 2 GPU cards.
- Process 5 requires 2 GPU cards.

☺ **Congratulations!**

You have completed today's challenging workshop!

The End of Workshop Project Guide