KNOWLEDGE ASSIMILATION AGENT USER MANUAL

WP5

Document Filename:	**KWF-WP5-D5-IISAS-v1.0-KAAUserManual.doc**
Work package:	**WP5**
Partner(s):	**II SAS**
Lead Partner:	**II SAS**
Document classification:	**PUBLIC**

Abstract: This document constitutes the user manual for the Knowledge Assimilation Agent tool which is part of K-WfGrid WP5.

Delivery Slip

	Name	Partner	Date	Signature
From	Zoltan Balogh	II SAS	07/10/2006	
Verified by	Piotr Nowakowski	CYFRONET	11/10/2006	
Approved by	Steffen Unger	FIRST	11/10/2006	

Document Log

Version	Date	Summary of changes	Author
0.4	09/30/2006	Document corrected according to internal review procedure	Zoltán Balogh
0.3	09/01/2006	Document extended.	Zoltán Balogh, Martin Šeleng, Martin Mališka
0.2	01/09/2006	Document corrected according to internal review procedure (KWF-IRF-D10.1-UIBK-v1.0-Wieczorek).	Zoltán Balogh
1.0	11/10/2006	QA check	Piotr Nowakowski

CONTENTS

COPYRIGHT NOTICE 4

1. INTRODUCTION..... 5

1.1. ABBREVIATIONS AND ACRONYMS 5
1.2. REFERENCES AND SOURCE CODE 5

2. SOFTWARE USAGE 7

2.1. KAA WEB SERVICE..... 7
2.1.1. Local hardware requirements 7
2.1.2. Local software requirements 7
2.1.3. Grid infrastructure requirements 7
2.1.4. Step-by-Step User Setup 7
2.1.5. Basic Operation..... 10
2.1.5.1. Request 10
2.1.5.2. Response 13
2.1.6. Advanced Features..... 14
2.1.7. Known Problems 14
KAA CLIENT..... 15
2.1.8. Local hardware requirements 15
2.1.9. Local software requirements 15
2.1.10. Grid infrastructure requirements 15
2.1.11. Step-by-Step User Setup 15
2.1.12. Basic Operation..... 15
2.2. KAA-GEMINI 15
2.2.1. Local hardware requirements 15
2.2.2. Local software requirements 15
2.2.3. Grid infrastructure requirements 15
2.2.4. Step-by-Step User Setup 15
2.2.5. Basic Operation..... 16
2.3. KAA WXA 16
2.3.1. Local hardware requirements 16
2.3.2. Local software requirements 16
2.3.3. Grid infrastructure requirements 16
2.3.4. Step-by-Step User Setup 16
2.3.5. Basic Operation..... 18
2.3.6. Advanced Features..... 18
2.3.7. Known Problems 19

3. INTERFACE REFERENCE GUIDE..... 20

4. TROUBLESHOOTING Q&A 22

4.1. KAA-WS 22

5. CONTACT INFORMATION AND CREDITS..... 23

6. THE EDG LICENSE AGREEMENT 24

COPYRIGHT NOTICE

This research is partly funded by the European Commission IST-2002-511385 Project “K-WfGrid”.

1. INTRODUCTION

Knowledge Assimilation Agent (KAA) is a knowledge-based component for Grid service workflows, which comprises three basic functionalities:

- assimilates run-time information about Grid services from different sources and produces performance estimations of future Grid service invocations (the KAA-Web Service);
- performs past workflow analysis and produces workflow result estimations (the KAA-WXA tool) and
- discovers new potential services through interactive semi-automatic ontology alignment (the OnTal tool).

KAA has several components which ensure its functionalities. The central part of KAA is the **KAA Web Service** (**KAA-WS**), which is a stateless web service implementation through which third party components request and retrieve the knowledge. KAA-WS implements the core learning algorithm, concretely an extended and improved instance-based learning technique. There is an internal ontology model of Grid service performances, resources and used invocation parameters which is called **Knowledge Assimilation Schema** (**KAS**). KAA consumes information generated by third party components such as Gemini (a monitoring infrastructure) and the Workflow Composition Tool (WCT) and transform them into KAS representation. New information are added upon completion of any grid workflow, which is detected by a component called **KAA-WCT**. Using a retrieved workflow identifier, a component called **KAA-Gemini** retrieves information about Grid services invocations and transforms them into KAS representation. **KAA-WXA** checks the GWES XML database for new workflows, gets the context from the workflow (based on XPath technique) and sends it to the UAA in a note format. If UAA has the same note (same context and input data) the old note is overwritten by the new one so users have always actual information about new results. **OnTal** follows one of general ideas of K-WfGrid, that broadening the knowledge of the system, stored in the form of semantic resources, allows more effective automatic workflow composition (performed by such components as WCT and AAB). Moreover, as the tool is semi-automatic, it is helpful for users who have to update the ontologies during services' registration.

In this deliverable we describe the tools which ensure the first two aspects (out of the three above mentioned functionalities) of KAA. The OnTal tool is described in a separate deliverable.

1.1. ABBREVIATIONS AND ACRONYMS

KAA	– Knowledge Assimilation Agent
KAS	– Knowledge Assimilation Schema
AAB	– Automatic Application Builder
GOM	– Grid Organizational Memory
WS	– Web Service
WSI	– Web Service Instance
WXA	– Workflow XML Analyzer

1.2. REFERENCES AND SOURCE CODE

Source code can be found in the project CVS repository. It can be either browsed through the CVS web interface:

```
http://cvs.ui.sav.sk/cgi-bin/cvsweb.cgi/kwfgrid/kaa/
```

or by directly checking out the kaa module from CVS:

```
cvs co :pserver:@cvs.ui.sav.sk:/home/cvs kwfgrid/kaa
```

Please note that for both ways of accessing the K-Wf Grid CVS repository a username and password is required.

Individual KAA components are placed in separate directories and can be checked out and installed separately. The following tools are placed in the kaa directory:

- **`kaa-client/`** - source code and libraries for a simple java command line client;
- **`kaa-gemini/`** - distribution of current KAA-GEMINI tool;
- **`kaa-webclient/`** - a simple web client distribution for KAA;
- **`kaa-ws/`** - KAA Web Service distribution;
- **`kaa-wxa/`** - KAA-WXA tool distribution.

Each of the above directories contains at least the following sub-directories:

- **`src/`** – tool source code directory;
- **`lib/`** – contains libraries required by the tool or by third party software utilized by the tool;
- **`build.xml`** – Apache Ant build file.

This KAA manual can be found in the K-Wf Grid BSCW server at:

`https://elbe.first.fhg.de/bscw/bscw.cgi/0/58119`

with file name:

`KWF-WP5-KAA-IISAS-v0.4-SoftwareUserManual.doc`

2. SOFTWARE USAGE

Bellow we provide some basic operating requirements by individual KAA tools.

2.1. KAA WEB SERVICE

2.1.1. Local hardware requirements

Hardware requirements for the machine where KAA would be deployed are:

- system architecture iX86 or other capable of running Java application,
- minimum 1 GHz processor (P4 or Athlon preferred),
- minimum 1 GB RAM (more is better),
- minimum 300 MB free disk space.

2.1.2. Local software requirements

Operating System requirements: UN*X system or derivate – Linux recommended (for prototype Mandrake 10.1 used).

In order to demonstrate KAA the following software is required (versions of respective software are only recommended and indicate the version which was used – other versions might but also might not work):

- Apache Tomcat (version 5.5.9)
- Apache Axis (version 1.2.1)
- Apache Ant

Java & Libraries required by KAA WS are:

- JDK 1.5 (jdk1.5.0_06 used for prototype),
- Jena (version 2.3 used for prototype),
- Castor (version 0.9.9 used for prototype).

2.1.3. Grid infrastructure requirements

KAA WS is a stateless web service which is running as an Axis application in a Tomcat container, what enables other services to connect to the service and consume its services. Therefore there is no need for special Grid infrastructure on the machine where KAA WS is deployed.

2.1.4. Step-by-Step User Setup

In order to install KAA WS please follow the following steps:

1. **Install JDK, Ant, Tomcat and Axis**. Test if installed correctly.

KAA assumes that `$CATALINA_HOME` is set. Further we assume the following configuration:

```
$JAVA_HOME=/usr/local/jdk
$CATALINA_HOME=/usr/local/tomcat
ANT_HOME=/usr/local/ant
```

The above directories are symbolic links:

```
tomcat -> jakarta-tomcat-5.5.9/
jdk -> jdk1.5.0_04/
ant -> apache-ant-1.6.2/
```

AXIS Installation Directory:

```
/usr/local/tomcat/webapps/axis
```

You need to make sure that your CLASSPATH is set correctly. Especially take great care of the following:

! Proper **`ant.jar`** must be in **`$CLASSPATH`**. During software testing we had to remove **`ant-1.5.1.jar`** from **`$CATALINA_HOME/webapps/axis/WEB-INF/lib`** because of a different version (we used version 1.6.2). During building **`$CLASSPATH`** in **`~/.bashrc`** it was imported earlier than **`$ANT_HOME/lib/ant.jar`** and that make ant not run.

2. **Checkout** KAA form CVS:

```
$ cvs co :pserver:@cvs.ui.sav.sk:/home/cvs kwfgrid/kaa/kaa-ws
```

3. **Build** KAA

Apache Ant is used to build the whole package. We are using some tool specific ant tasks, so Ant must see some Java libraries:

- In order to communicate with Tomcat Manager
  `$CATALINA_HOME/server/lib/catalina-ant.jar`
  must be copied into the lib directory of your Ant installation.
- You will also need to copy
  `$CATALINA_HOME/webapps/axis/WEB-INF/lib/axis-ant.jar`
  into lib directory of Ant.

Change directory to:

```
$ cd kwfgrid/kaa/kaa-ws
```

You should have a build.xml file in the current directory:

```
$ ls -l
...
-rw-r--r--  1 zb pdc  7321 Nov  7 13:20 build.xml
...
```

Build KAA:

```
$ ant
```

Default Ant task is to build a jar package of the project, so the above command is equivalent to:

```
$ ant jar
```

4. **Start Apache Tomcat** (we assume that Axis is already installed).

Must be a super-user (root) or owner of Tomcat installation to do this:

```
$ $CATALINA_HOME/bin/catalina.sh start
```

Check in a web browser if Tomcat/Axis is running. You should be able to access these URLs:

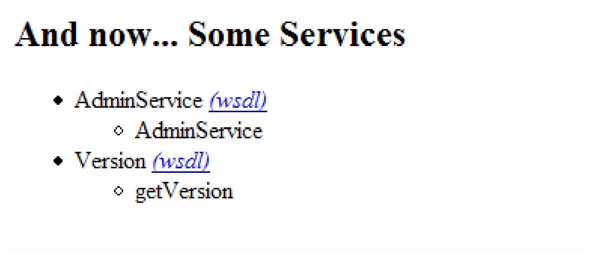
```
http://your.host:8080/
http://your.host:8080/axis/
```

5. **Check the list of applications deployed in Axis**

The list of applications deployed in Axis can be seen by submitting this query:

```
http://your.host:8080/axis/servlet/AxisServlet
```

You should see something like this:



And now... Some Services

- AdminService (wsdl)
  - AdminService
- Version (wsdl)
  - getVersion

Make sure that KAA WS is not displayed on the list. If KAA WS is deployed (because of any previous installations), undeploy it first:

```
$ ant undeploy-kaaws
```

Undeploying KAA WS is necessary because you can have some older versions of web services deployed, so we clean the Axis before deploying our new version.

6. **Restart Tomcat/Axis**

Axis can be restarted as following (we recommend to open a separate console where Tomcat restarting will be managed) from the **`$CATALINA_HOME`** directory:

```
$ bin/catalina.sh stop; bin/catalina.sh start
```

or

```
$ bin/shutdown.sh; bin/startup.sh
```

7. **Configure properties**

Tomcat directory is extracted from shell environment variable called **`$CATALINA_HOME`**., so user does not need to set it in property file, just needs to make sure that his current environment variable is set correctly.

All the settings required by KAA WS should be made in file:

```
kwfgrid/kaa/kaa-ws/src/net/kwfgrid/kaa/config/kaa.properties
```

The following properties are pre-set:

```
KAA_DATA_DIR=/var/local/kwfgrid/kaa-ws
```

After setting the **`KAA_DATA_DIR`** please make sure that the set directory is writable for the current user. Initial case base will be copied to this directory.

8. **Deploy KAA WS**

Before deploying KAA WS make sure that:

- Tomcat is up and running and that
- you have write permission to directory **`$CATALINA_HOME/webapps/axis/WEB-INF`**

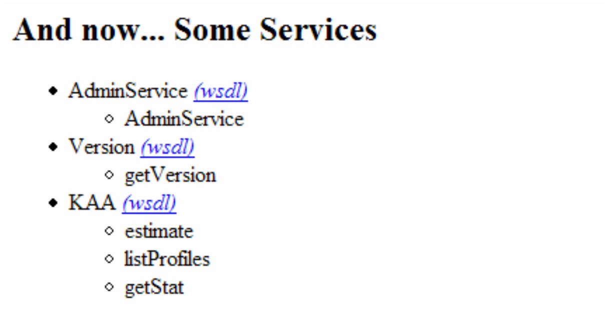
Deploy KAA:

```
$ ant deploy-kaaws
```

Deployment should not display any errors. If there are problems with any permission, ant will complain about it. Restart Tomcat after KAA WS is deployed.

9. **Check if the KAA WS is running**

By listing the deployed Axis applications you should see something like this:



And now... Some Services

- AdminService (wsdl)
  - AdminService
- Version (wsdl)
  - getVersion
- KAA (wsdl)
  - estimate
  - listProfiles
  - getStat

If you see such output, it means you have successfully deployed KAA WS to Axis and you can start using it.

2.1.5. Basic Operation

2.1.5.1. Request

The KAA Web Service implements the following basic operations:

- **`estimate(String request)`** – estimates a result based on the valid provided request;
- **`listProfiles()`** – lists all profiles known to KAA;
- **`getStat()`** – returns statistics about the KAA case base.

In order to have KAA respond correctly, a client must send a properly formed request. Request is a regular XML compliant with this XSD schema:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

```
              targetNamespace="http://net.kwfgrid/kaa">

     <xsd:annotation>
       <xsd:documentation xml:lang="en">
        Data schema for KAA request.
        Version 0.2
        Created by: Zoltan Balogh, balogh@savba.sk
       </xsd:documentation>
     </xsd:annotation>

     <xsd:element name="request" type="Request"  />

     <xsd:complexType name="Request">
       <xsd:sequence>
         <xsd:element name="features" type="Features"/>
         <xsd:element name="wscontext" type="WSContext"/>
         <xsd:element name=”result” type=”Result”/>
         <xsd:element name="wsalt" type="WSAlt"/>
       </xsd:sequence>
     </xsd:complexType>

     <xsd:complexType name="Features">
       <xsd:sequence>
         <xsd:element name="param" type="Param" minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
     </xsd:complexType>

     <xsd:complexType name="Param">
       <xsd:attribute name="name" type="xsd:string"/>
       <xsd:attribute name="value" type="xsd:string"/>
     </xsd:complexType>

     <xsd:complexType name="WSContext">
       <xsd:attribute name="class" type="xsd:string"/>
       <xsd:attribute name="operation" type="xsd:string"/>
     </xsd:complexType>

     <xsd:complexType name=”Result”>
       <xsd:attribute name="concept" type="xsd:string"/>
     </xsd:complexType>

     <xsd:complexType name="WSAlt">
       <xsd:sequence>
         <xsd:element name="ws" type="WS" minOccurs="1" maxOccurs="unbounded"/>
       </xsd:sequence>
     </xsd:complexType>

     <xsd:complexType name="WS">
       <xsd:attribute name="uri" type="xsd:string"/>
     </xsd:complexType>

   </xsd:schema>
```

Sample requests are the following:

```
<?xml version="1.0"?>
<request>
 <features>
   <param name="temp" value="35"/>
 </features>

 <wscontext class="NetFileParser" operation="computeCtmStartZonePolyg"/>

 <result concept=”WsOperationRunTime”/>
 <wsalt>
   <ws uri="http://grid02.softeco.it/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser"/>
   <ws uri="http://kwfgrid1.dps.uibk.ac.at/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser"/>
 </wsalt>
</request>
```

```
<?xml version="1.0"?>
<request>
 <features>
   <param
      name="hasWsDeployment.hasURI.100"
     value="http://kwfgrid1.dps.uibk.ac.at/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser"/>
 </features>

 <wscontext class="NetFileParser" operation="computeCtmStartZonePolyg"/>

 <result concept=”WsOperationInvocationSuccess”/>

 <wsalt>
   <ws uri="http://grid02.softeco.it/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser"/>
   <ws uri="http://kwfgrid1.dps.uibk.ac.at/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser"/>
 </wsalt>
</request>
```

```
<?xml version="1.0"?>
<request>
  <features>
    <param name="hasNS.10" value="2"/>
    <param name="hasEW.10" value="4"/>
    <param name="hasPROC.20" value="4"/>
    <param name="hasSIM_T.60" value="360"/>
  </features>

  <wscontext class="MM5" operation="run"/>

  <result concept=”WsOperationRunTime”/>

  <wsalt>
    <ws uri="http://cluster.ui.sav.sk/mm5"/>
    <ws uri="http://cluster.cyfronet.pl/service/mm5"/>
    <ws uri="http://cluster.softeco.it/service/mm5"/>
  </wsalt>
</request>
```

The composition of individual features is described in the Developer Manual.

Explanation of the tags:

- **`<features>`** - contain features to be used;
- **`<wscontext class`** - class of the WS for which we want to estimate the run-time;
- **`<wscontext operation`** - WS operation for which we estimate the runtime;
- **`<result concept`** – Concept of a result which specifies the result to be predicted;

- **`<wsalt>`** - do not consider all deployments of a WS of concrete class but consider only some specific ones (must be listed)

2.1.5.2. Response

Response is a regular XML compliant with this XSD schema:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://net.kwfgrid/kaa">

  <xsd:annotation>
    <xsd:documentation xml:lang="en">
     Data schema for KAA response.
     Version 0.2
     Created by: Zoltan Balogh, balogh@savba.sk
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="result" type="Result" />

  <xsd:complexType name="Result">
    <xsd:sequence>
      <xsd:element name="WsOperationRunTime" type="ResultType" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element name="WsOperationInvocationSuccess" type="ResultType" minOccurs="0"
        maxOccurs="unbounded"/>
      <xsd:element name="value" type="xsd:string"/>
      <xsd:element name="status" type="xsd:string"/>
      <xsd:element name=”message” type=”xsd:string”/>
      <xsd:element name="count" type="xsd:string"/>
      <xsd:element name="cases" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="ResultType">
    <xsd:attribute name="wsuri" type="xsd:string"/>
  </xsd:complexType>

</xsd:schema>
```

KAA responds with the following message (example):

```
<?xml version="1.0"?>
<result>
 <WsOperationRunTime wsuri="http://grid02.softeco.it/cgi-
bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser">
   <value>2034.0</value>
   <status>OK</status>
   <message></message>
   <count>2</count>
 </runTime>
 <WsOperationRunTime wsuri=" http://kwfgrid1.dps.uibk.ac.at/cgi-
bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser">
   <value>-1.0</value>
   <status>OK</status>
   <message></message>
   <count>2</count>
 </runTime>
</result>
```

```
<?xml version="1.0"?>
<result>
 <WsOperationInvocationSuccess wsuri="http://grid02.softeco.it/cgi-
bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser">
   <value>0.85</value>
   <status>OK</status>
   <message></message>
   <count>2</count>
 </runTime>
 <WsOperationInvocationSuccess wsuri=" http://kwfgrid1.dps.uibk.ac.at/cgi-
bin/SOAP.cgi/net/kwfgrid/ctm/NetFileParser">
   <value>-1.0</value>
   <status>OK</status>
   <message></message>
   <count>2</count>
 </runTime>
</result>
```

Explanation of the tags:

- **`<result>`** - basic tag which contains the overall result generated;
- **`<WsOperationRunTime>`** - contains one type of result, in this case length of WS operation run time;
- **`<WsOperationInvocationContext>`** - contains one type of result, in this case success or failure of WS operation invocation;
- **`<value>`** - value of the result;
- **`<status>`** - status of the result;
- **`<message>`** - explanatory result message;
- **`<cases>`** - how many cases were used to estimate the current type of the result;
- **`<count>`** - wsalt serial number.

2.1.6. Advanced Features

No advanced features to describe.

2.1.7. Known Problems

No known issues.

KAA CLIENT

There is a simple KAA shell client using generic Axis Client (no need to deal with stubs generated with Axis).

2.1.8. Local hardware requirements

For the KAA shell clients a general network-connected workstation capable of running Java is sufficient.

2.1.9. Local software requirements

Install JDK and Ant as described in KAA WS software requirements.

2.1.10. Grid infrastructure requirements

None.

2.1.11. Step-by-Step User Setup

Here are the steps to install the KAA client:

```
$ cvs co kwfgrid/kaa/kaa-client

$ cd kwfgrid/kaa/kaa-client/

$ ant

$ chmod u+x ./kaa-client.sh

$ ./kaa-client.sh
```

2.1.12. Basic Operation

Usage is:

```
./kaa-client.sh WS_URI WS_OPERATION XML_REQUEST
```

If no arguments are provided, it has a sample request built in, which is submitted to KAA. Default KAA is deployed at cvs.ui.sav.sk at the following URI:

```
http://cvs.ui.sav.sk:8080/axis/services/KAA
```

2.2. KAA-GEMINI

2.2.1. Local hardware requirements

Hardware requirements are the same as for the KAA web service described in section 2.1.2

2.2.2. Local software requirements

Install JDK and Ant as described in KAA WS software requirements, all 3-rd party libraries are stored in CVS and will be retrieved during the setup of the component.

2.2.3. Grid infrastructure requirements

None.

2.2.4. Step-by-Step User Setup

To build KAA Gemini library, use the following steps:

```
$ cvs co kwfgrid/kaa/kaa-gemini

$ cd kwfgrid/kaa/kaa-gemini/

Edit URI address of the monitoring service in the file
src/net/kwfgrid/kaa/gemini/monitoring.properties

$ ant dist
```

To install KAA Gemini into the KAA WXA module, just simply copy all libraries from directory “dist” to the directory where KAA WXA libraries are installed.

2.2.5. Basic Operation

Here is example how to use KAA Gemini module:

```
String workflowId = "truong_96e9e650-1669-11db-9222-8b8a92131fb0";

EventProcessor eventProcessor = new EventProcessor();

eventProcessor.processEvents(workflowId);

OwlWriter writer = new OwlWriter();


writer.setActivityIterator(eventProcessor.getActivityMap().values().itera
tor());

writer.write();
```

2.3. KAA WXA

2.3.1. Local hardware requirements

Same as KAA-WS.

2.3.2. Local software requirements

Same as KAA-WS.

2.3.3. Grid infrastructure requirements

None

2.3.4. Step-by-Step User Setup

1. Checkout WXA frOm CVS:

```
$ cvs co :pserver:@cvs.ui.sav.sk:/home/cvs kwfgrid/kaa/kaa-wxa
```

2. Configure WXA

Change directory to:

```
$ cd kwfgrid/kaa/kaa-wxa
```

You should see a **build.properties** file, which contains following properties with default values:

```
build=classes
```

```
dist=/usr/local/kaa-wxa/
src=src
```

Only the **`dist`** properties can be changed. Do not modify the other properties. The **dist** property determines directory where WXA component will be installed.

Change directory to:

```
$ cd config
```

These directory contains three properties files:

```
DBWorkflows.properties
wxa.properties
wxaxpath.properties
```

The **`DBWorkflows.properties`** file contains properties for connecting and retrieving past workflows from **Exist** database (please refer to GWES):

```
DB_URL=http://fhrg.first.fraunhofer.de:8080/exist/webdav
SERVICE_URL=http://fhrg.first.fraunhofer.de:8080/exist/services/Query
DB_USER=gwes
DB_PASSWD=sewg!
XQUERY=collection('/db/gworkflowdl')
DB_WORKFLOWS_PATH=/usr/local/kaa-wxa/
```

Below is the description of properties and values in **`DBWorkflows.properties`** file:

`DB_URL` is URL of **Exist** service (please refer to GWES)

`SERVICE_URL` is URL of **Exist** web service (please refer to GWES)

`DB_USER` is a name of a user (please refer to GWES)

`DB_PASSWD` is a password for DB_USER (please refer to GWES)

`XQUERY` is a relative xpath ('db/gworkflowdl')for querying past workflows from **Exist** database (please refer to GWES)

`DB_WORKFLOWS_PATH` is a path for **`DBWorkflowsID.txt`** file. It is a file to which WXA downloads workflows from XML database for further processing. Please set this property to the same value as **dist** property in **`build.properties`** file.

`wxa.properties` file contains basic properties for WXA component:

```
NS1=gw=http://www.gridworkflow.org/gworkflowdl
NS2=ws=http://www.gridworkflow.org/gworkflowdl/wsclassoperation
PORTAL_URL=http://portal.ui.sav.sk:8000
EMBET_USER_NAME=kaa-wxa
```

`NS1` and **`NS2`** properties are namespace properties for xml schemas used in workflow xml file. Please do not modify these properties.

`PORTAL_URL` is URL of deployed UAA (please refer to UAA)

`EMBET_USER_NAME` is a name which appears in UAA as a name of the author of notes added by WXA.

3. Build WXA

Build KAA-WXA:

Return back to the **`kwfgrid/kaa/kaa-wxa`** directory and run Ant task.

```
$ ant
```

Default Ant task is to build a jar package of the project, so the above command is equivalent to:

```
$ ant dist
```

If the Ant task is completed with no errors, WXA component is installed successfully.

4. Check the **`wxa.jar`** file (OPTIONAL)

Please change the directory to the one set in **dist** property (see **Configure wxa**). You can see something like this:

```
$ ls -l
drwxrwxr-x    4    kwfgrid pdc     12245622 Aug 29  2006 lib/
-rwxrwxr-x    1    kwfgrid pdc        21499 Aug 29 14:49 kaa-wxa.jar
```

2.3.5. Basic Operation

Change the directory to **`/etc`**. Add the following line to the **`crontab`** file like this:

```
*/5****  java -jar {dist}/kaa-wxa.jar
```

Where the first part */5**** represents frequency of running the WXA component (Value 5 means running WXA every 5 minutes. This value can be changed.) to check and process new workflows in workflow xml database. **{dist}** part of the path is the value of **dist** property (see **Configure WXA**).

To launch WXA manually please only call the **`kaa-wxa.jar`** (start WXA component). Everything the **`kaa-wxa.jar`** needed is in the properties files inside the jar file.

```
$ java -jar {dist}/kaa-wxa.jar
```

2.3.6. Advanced Features

`wxapath.properties` file contains XPath queries and other useful properties (described below) for getting data from past workflows:

```
…
STATUS_PATTERN=/*/*[@name='status']/text()
STATUS_TEXT=!

OWL1_DATA_PATTERN=/*/*[@ID='generateSVGFile']
OWL1_DATA_CONTEXT=#serviceCTM_NetFileParser
OWL1_DATA_CLASS=CLASS
```

```
OWL2_DATA_PATTERN=/*/*[@ID='getSVGFileURL']
OWL2_DATA_CONTEXT=#serviceCTM_NetFileParser
OWL2_DATA_CLASS=CLASS

START_ZONE_PATTERN=/*/*[@ID='startZoneId']/*/*/*/text()
START_ZONE_TEXT=Start Zone!

END_ZONE_PATTERN=/*/*[@ID='endZoneId']/*/*/*/text()
END_ZONE_TEXT=End Zone!

RESULT_PATHSVG_DATA_PATTERN=/*/*[@ID='PathSVG']/*/*/*/text()

RESULT_EMISSIONSVG_DATA_PATTERN=/*/*[@ID='EmissionSVG']/*/*/*/text()
…
```

These properties can be divided into four categories:

- **PATTERN** is a query for data of the workflow
- **TEXT** is the text which appears in the label of the note in UAA. The exclamation mark (!) is a symbol for data retrieved by the query
- **CONTEXT** is the context of the note (please refer to UAA)
- **CLASS** is only for queries in service like Air Pollution Calculator. This service first use Net File Parser service and then calculates the emissions (please refer to CTM application), so WXA don't know the context because he found two different queries in one workflow (the CLASS property must have a CLASS value). For result which is URL address there is only **PATTERN** part of the properties. Properties which are grouped together (**PATTERN**, **TEXT**, **CONTEXT** and **CLASS**) must have the same name. For example:

  ```
  XXX_PATTERN
  XXX_TEXT
  XXX_CONTEXT
  XXX_CLASS
  ```

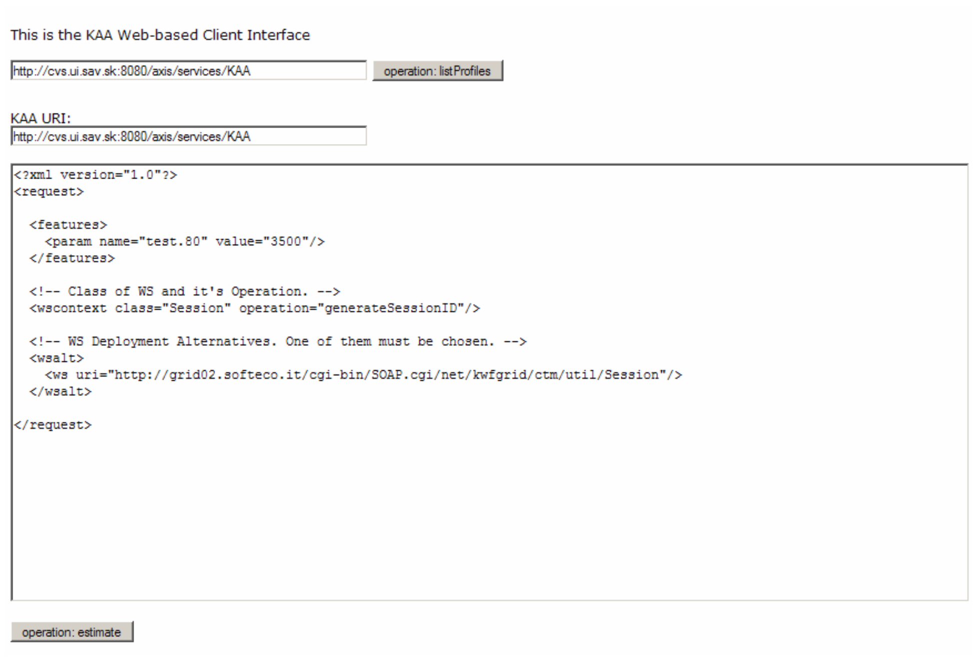
! Wrong queries can malfunction the WXA component. Please modify carefully.

2.3.7. Known Problems

No issues yet.

3. INTERFACE REFERENCE GUIDE

Prototype was developed as a portlet component to the K-Wf Grid GridSphere portal. The interface allows submitting a query to a KAA deployment.


This is the KAA Web-based Client Interface

http://cvs.ui.sav.sk:8080/axis/services/KAA

KAA URI:
http://cvs.ui.sav.sk:8080/axis/services/KAA

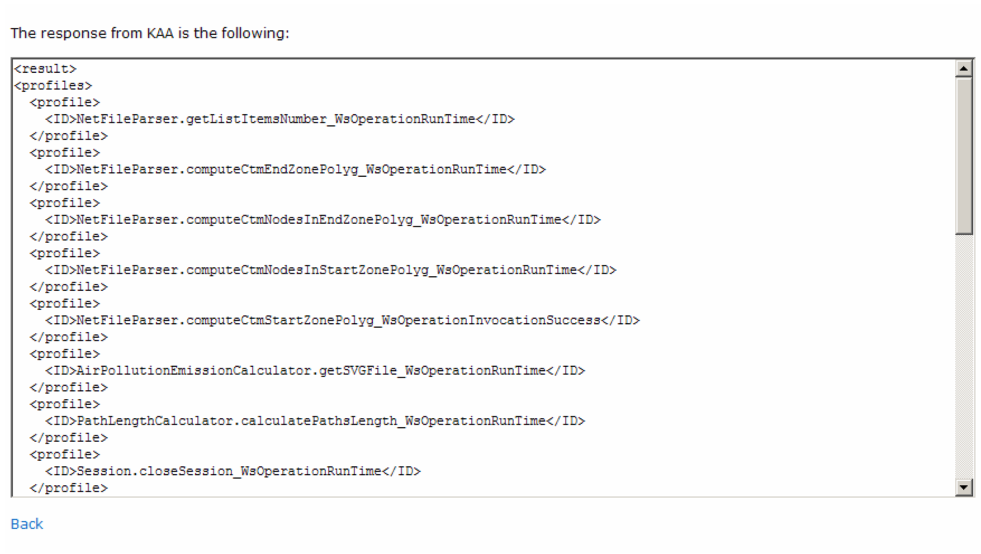
```
<?xml version="1.0"?>
<request>

  <features>
    <param name="test.80" value="3500"/>
  </features>

  <!-- Class of WS and it's Operation. -->
  <wscontext class="Session" operation="generateSessionID"/>

  <!-- WS Deployment Alternatives. One of them must be chosen. -->
  <wsalt>
    <ws uri="http://grid02.softeco.it/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/util/Session"/>
  </wsalt>

</request>
```

Fig. 3-1 KAA Input dialog

The KAA web interface allows:

- setting the URL of the KAA WS and
- formulating an XML request to KAA.

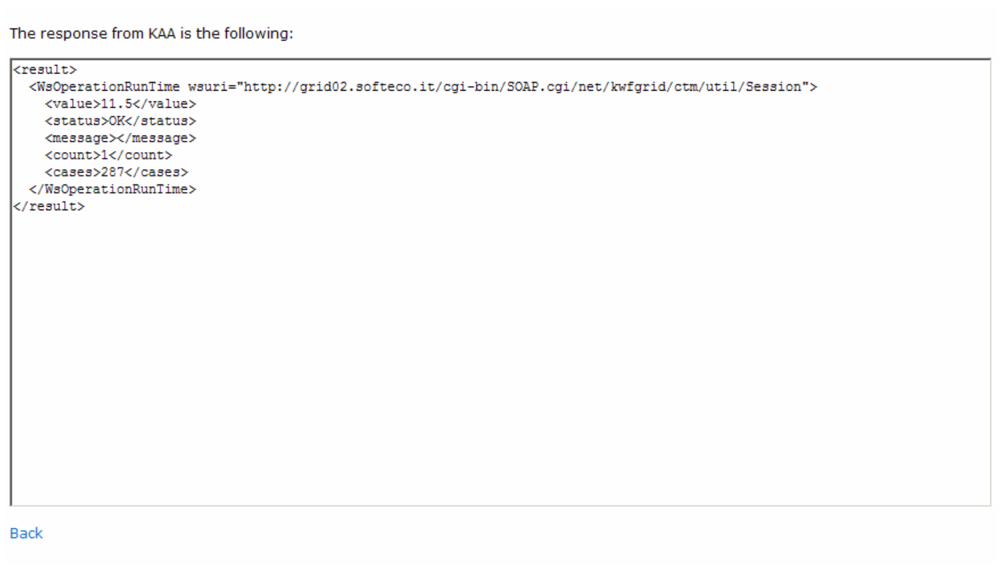
The response to the listProfiles() operation is provided in a textbox:


The response from KAA is the following:

```
<result>
<profiles>
  <profile>
    <ID>NetFileParser.getListItemsNumber_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>NetFileParser.computeCtmEndZonePolyg_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>NetFileParser.computeCtmNodesInEndZonePolyg_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>NetFileParser.computeCtmNodesInStartZonePolyg_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>NetFileParser.computeCtmStartZonePolyg_WsOperationInvocationSuccess</ID>
  </profile>
  <profile>
    <ID>AirPollutionEmissionCalculator.getSVGFile_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>PathLengthCalculator.calculatePathsLength_WsOperationRunTime</ID>
  </profile>
  <profile>
    <ID>Session.closeSession_WsOperationRunTime</ID>
  </profile>
```

Fig. 3-2 KAA Response to the listProfiles() operation.

The output of KAA is returned as the XML in a textbox.


The response from KAA is the following:

```
<result>
  <WsOperationRunTime wsuri="http://grid02.softeco.it/cgi-bin/SOAP.cgi/net/kwfgrid/ctm/util/Session">
    <value>11.5</value>
    <status>OK</status>
    <message></message>
    <count>1</count>
    <cases>287</cases>
  </WsOperationRunTime>
</result>
```

Fig. 3-3 KAA Response to the estimate() operation.

The above response shows result to the estimate() operation.

4. TROUBLESHOOTING Q&A

Answers to known problems and frequently asked questions are provided below sorted according to individual tools:

4.1. KAA-WS

Q: Why is KAA supporting only the MM5 WSBPP?

A: For the purpose of first prototype only MM5 WSBPP was created and data for such profile were collected. The first prototype only demonstrates the capability of KAA.

Q: When I submit a request to KAA through the web interface I get no response. What is the problem?

A: Please make sure that the KAA web service is running at the provided location or check if the URI you have typed is correct.

Q: When I try to launch KAA it returns a Java exception. What is the problem?

A: Problem is most probably in non compatible libraries in Axis. KAA uses different third party software such as JENA or Castor which require certain versions of libraries. If other libraries will be provided, KAA will fail.

Q: When I submit a query to KAA I get an "java.lang.NullPointerException". Where is the problem?

A: Most probably Axis can not reach one or some of the required libraries (*.jar). Please check your **`$CATALINA_HOME/webapps/axis/WEB-INF/lib`** if all required libraries are present. KAA especially requires the following ones:

- **`kwfgrid-kaa.jar`** – of course we need this! Otherwise there is no way to make KAA work.
- **`Gomclient.jar`** – this makes it possible to access the K-Wf Grid GOM.
- **`castor-0.9.9.jar`**
- **`jena.jar`**
- **`log4j-1.2.8.jar`**
- **`xercesImpl.jar`** – Caution!!! Jena requires different version of xercesImpl.jar than the one Axis has by default! Therefore it must be overwritten by the correct one.

5. CONTACT INFORMATION AND CREDITS

Zoltán Balogh, balogh@savba.sk, II SAS Bratislava, Slovakia
Martin Šeleng, martin.seleng@savba.sk, II SAS Bratislava, Slovakia
Martin Mališka, martin.maliska@savba.sk, II SAS Bratislava, Slovakia

6. THE EDG LICENSE AGREEMENT