LarKC

The Large Knowledge Collider: a platform for large scale integrated reasoning and Web-search

FP7 – 215535

Hand-out materials for the EARLY ADOPTERS Workshop

Coordinator: Matthias Assel, HLRS
With contributions from: Alexey Cheptsov, HLRS; Luka Bradesko, Blaz Fortuna, CycEur; Spyros Kotoulas, VUA.

<table>
<thead>
<tr>
<th>Document Identifier:</th>
<th>LarKC/2010/EAT /v1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Deliverable:</td>
<td>-</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>3.11.2010</td>
</tr>
<tr>
<td>State:</td>
<td>Final</td>
</tr>
<tr>
<td>Distribution:</td>
<td>Public</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

The Large Knowledge Collider (LarKC) is a platform for building efficient reasoning applications for the Semantic Web. In the tutorial, the participants will be offered a set of exercises and hands-on sessions, leading them through all the necessary steps towards building their own applications with support of LarKC. The tutorial is organized in four sessions.

In **Hands-on 1**, the participants will get introduction to LarKC and execute an existing workflow based application using the LarKC platform.

In **Hands-on 2**, the participants will be asked to enhance the workflow used in the previous exercise.

In **Hands-on 3**, the participants will develop a new workflow component (plug-in) and integrate it with the workflow used in the previous exercise.

In **Hands-on 4**, the participants will extend the workflow from the previous exercise with a parallelized plug-in, therefore improving the workflow efficiency.

This document acts as a hand-out material, guiding the participants step-by-step through all the necessary steps needed for successful performing the exercises.

All the materials needed for the exercises as well software components are available on the distributed usb flash sticks or at the link at sourceforge.

The content on the usb flash sticks, distributed among the tutorial participants, has the following structure:

```
/ LarKC
  / platform
  / workflows
  / development_kit – Eclipse wizard for development new plug-ins
  / sample_data – rdf data needed for the tutorial
```
**TABLE OF CONTENTS**

1. HANDS-ON SESSION 1. DEPLOYING LARKC AND RUNNING A SIMPLE WORKFLOW ........................................................................................................................................ 4

2. HANDS-ON SESSION 2. ENHANCEMENT OF THE EXISTING WORKFLOW........... 7

3. HANDS-ON SESSION 3. DEVELOPMENT OF A NEW PLUG-IN AND USING IT THE WORKFLOW ......................................................................................................................................................... 8

4. HANDS-ON SESSION 4. USING PARALLELISED PLUG-INS......................... 15

5. ADDITIONAL INFORMATION ABOUT LARKC.............................................. 16
1. Hands-on session 1. Deploying LarKC and running a simple workflow

Session goals:
- get familiar with the LarKC platform;
- execute an existing workflow based application with LarKC;
- set up LarKC in Eclipse IDE;

Outcome:
- the deployed LarKC platform on the participants’ computers;
- a running reasoning application with LarKC.

Steps to be done:
I. Deployment of LarKC on the user’s local machine
- copy the content of the usb stick to the local directory, e.g. “larkc”

II. Running LarKC
- change to platform folder and execute run-larkc command.

In case of successful platform start, there should be the following output produced:

```
INFO: Starting the internal HTTP server on port 8182
"Initial Lisp Listener" 01:18:48.579 INFO e.l.c.m.ManagementInterfaceMain: Management server started on 8182
```

III. Executing an existing workflow
- Start an html browser
- Open file platform/larkc.html in the browser
IV. Executing a workflow

In this exercise we ask you to execute the following workflow:
The workflow performs reasoning over RDF triples stored locally (included in the content of the LarKC usb flash stick as well).

- open the file with the workflow description for the first exercise (`workflows/EAW_Workflow1.rdf`) in any of text editors
- copy the xml-based workflow description to the “Workflow Description” window of larkc.html
- click on “Submit” button below the “Workflow Description” window
- copy the SPARQL query to the “SPARQL-Query” window of larkc.html
- click on “Submit” button below the “SPARQL-Query” window
- store the output produced by the browser in any text file (e.g. `Workflow1_Output.txt`)
2. Hands-on session 2. Enhancement of the existing workflow

Session goals:
- get familiar with the LarKC workflow description
- construct new workflows with LarKC

Outcome:
- improved reasoning application from the previous exercise

In this session, you will increase the results of the reasoning, done in the previous exercise, by getting additional data from the Web using a Semantic Web search engine (Sindice). The workflow will be extended as shown below:

Steps to be done:

I. Submit a workflow from “EAW_Workflow2.rdf” to the LarKC management interface
II. Submit a query from “EAW_Workflow2.rdf” to the LarKC management interface
III. Store the output
3. Hands-on session 3. Developing a new plug-in and using it the workflow

Session goals:
- familiarise yourself with the plug-in development technique for LarKC
- develop new plug-in
- add new plug-in to a workflow

Outcome:
- improved reasoning application from the previous exercise

Steps to be done:

I. Install the plug-in wizard to your Eclipse installation
   - Copy the
     `development_kit/eclipse_wizard/LarKCWizard_1.0.0.201011111832.jar`
     to the `/plugins/` subdirectory of your Eclipse Installation.
   - Run Eclipse
   - Create new workspace

   ![Workspace Launcher](image)

   - Make sure that, in the Eclipse toolbar, there is a new LarKC icon
II. Use the wizard to create a new empty plug-in
   ▪ Select the Other option from the File → New → Other menu
- Select the LarKC plugin option from LarKC directory

- In the following dialog enter the name of your new plug-in and point to the location of your local LarKC platform installation
III. Write the code for the new plug-in
   - Open the code for MyLarkePlugin project as shown in the figure

   - Populate the plug-in with the code, which creates and returns a new set of statements (invokeInternal method):

```java
// the plug-in code
ArrayList<Statement> statements = new ArrayList<Statement>();
// RDF Resources
URI meURI = new URIImpl("http://example.com/me");
URI frankURI = new URIImpl("http://dblp.l3s.de/d2r/resource/authors/Frank_van_Harmelen");
URI foafPersonURI = new URIImpl("http://xmlns.com/foaf/0.1/Person");
URI foafNameURI = new URIImpl("http://xmlns.com/foaf/0.1/name");
URI foafKnowsURI = new URIImpl("http://xmlns.com/foaf/0.1/knows");
// RDF statements
statements.add(new StatementImpl(meURI, RDFConstants.RDF_TYPE, foafPersonURI));
statements.add(new StatementImpl(meURI, foafNameURI, new LiteralImpl("My Name")));
statements.add(new StatementImpl(frankURI, RDFConstants.RDF_TYPE, foafPersonURI));
statements.add(new StatementImpl(frankURI, foafNameURI, new LiteralImpl("Frank van Harmelen")));
statements.add(new StatementImpl(meURI, foafKnowsURI, frankURI));
// create set of statements and return it as a result
URI resultGraphName = new URIImpl("http://example.com/result");
return DataFactory.INSTANCE.createRdfGraph(statements, resultGraphName);
```
- Start the platform by right-clicking on the project and selecting Run As → Java Application

- Using the search button, Search for and select eu.larkc.core.Larkc as the main class.
- Under Arguments tab, point Working directory to the installed platform.
IV. Extend the Workflow from the previous exercise, adding the developed plug-in

- Open larkc.html in the browser.
- Try *EAW-Workflow3-A.rdf* and test SPARQL query
- Try *EAW-Workflow3-B.rdf* and test SPARQL query

In this hands-on you will include some parallelised plug-ins in the workflow. The WebPIE engine, consisting of three plug-ins (to perform dictionary encoding, to calculate the OWL Horst closure and to perform dictionary decoding) is used to perform materialization of the input in parallel. Then irrelevant data from the files is excluded, by filtering on the triple patterns that appear in the query. Finally the results are included in the reasoner, as in the previous workflow.

Session goals:
- to get knowledge of using parallelised plug-ins in the workflow
- improve performance by filtering out irrelevant data

Outcome:
- improved reasoning application from the previous exercise

Steps to be done:
I. Use EAW-workflow4.xml
II. Post the query from the previous examples
III. Get the results
5. Additional information about LarKC