

Charles University in Prague
Faculty of Mathematics and Physics

ODCleanStore

Linked Data management tool

Administrator's & Installation Manual

Release 1.0
March 16, 2013

Authors: Jan Michelfeit
Dušan Rychnovský
Jakub Daniel
Petr Jerman
Tomáš Soukup

Supervisor: RNDr. Tomáš Knap

Contents

1	Introduction	2
2	System Requirements	3
2.1	OpenLink Virtuoso	3
2.1.1	Creating Database Instances	3
3	Step by Step Installation Guide	5
3.1	Prerequisites	5
3.2	Using the Installer	5
3.3	Post-installation Steps	9
4	Starting and Stopping the Application	10
4.1	Engine	10
4.2	Administration Frontend	10
5	Trying out ODCleanStore	11
6	Configuration Options	13
7	Custom Transformers	18
A	Distribution Contents	19
B	License	22

1. Introduction

ODCleanStore is a server application for management of Linked Data. It stores data in RDF, processes them and provides integrated views on the data.

This document serves as the installation guide and manual for system administrators. It includes system requirements, installation instructions, configuration overview, manual how to start the application and basic information about extending ODCleanStore's data processing capabilities by adding *custom transformers*.

What is ODCleanStore

ODCleanStore accepts arbitrary RDF data and metadata through a SOAP webservice (*Input Webservice*). The data is processed by *transformers* in one of a set of customizable *pipelines* and stored to a persistent store (OpenLink Virtuoso database instance). The stored data can be accessed again either directly through a SPARQL endpoint or through *Output Webservice*. Linked Data consumers can send queries and custom query policies to Output Webservice and receive (aggregated/integrated) RDF data relevant for their query, together with information about provenance and data quality.

Related documents

More detailed information about ODCleanStore from the perspective of a user can be found in related document "User Manual". It also contains definition of user roles, glossary of terms etc. Document "Programmer's Guide" contains more detailed information intended primarily for developers, system administrators, however, may also find interesting in-depth information in Programmer's Guide.

2. System Requirements

ODCleanStore is a multiplatform application written in Java. These are the system requirements for installing and running ODCleanStore:

- Operating System: Windows XP, Windows 7, Windows Server 2008, Linux
- Java Runtime Environment 6 or newer¹
- Installed OpenLink Virtuoso² (see below)
- Apache Tomcat³ (or an equivalent Java servlet container)

2.1 OpenLink Virtuoso

ODCleanStore uses two instances of Virtuoso database for storing data – one is used for storing data that are currently being processed (*dirty database instance*) and the other for already processed data visible to end users (*clean database instance*).

ODCleanStore requires that the two instances of Virtuoso database are created before the installation process. If you are familiar with Virtuoso and already have two Virtuoso instances that you would like to use, you can skip the rest of this section – make sure, however, to modify ODCleanStore configuration options accordingly (see Chapter 6). In particular, set correct database settings and include a directory specified in `engine.clean_import_export_dir` and `engine.dirty_import_export_dir` configuration options in `DirsAllowed` option of Virtuoso configuration. The user account which is used to run Engine must have permissions to both read and write in the directory.

2.1.1 Creating Database Instances

Virtuoso has a paid enterprise license, but also an open source edition⁴ available for free. In order to install it, follow instructions on the download page. Make sure that the deployed binaries (`<installation-directory>/bin`) are on your system PATH. You can test it by running commands `isql` (check that it is the Virtuoso `isql`) and `virtuoso-t`. If `isql` cannot be on PATH for any reason, pass the full path to the executable as a parametr of the installer.

ODCleanStore distribution contains files `virtuoso.ini-clean` and `virtuoso.ini-dirty`. While it is not required, we strongly recommend creating the two database instances with these provided configuration files.

1. Create one empty directory for each database instance.
2. Copy `virtuoso.ini-clean` to the directory for the clean database instance, create a copy of the file and name it `virtuoso.ini`. If you are familiar with Virtuoso, you may customize configuration in `virtuoso.ini` according to your needs.
3. Create a new database instance by executing

```
virtuoso-t +service create +instance odcs-clean +configfile virtuoso.ini
```

¹<http://www.java.com/en/download/>

²<http://virtuoso.openlinksw.com/>

³<http://tomcat.apache.org/>

⁴<http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSDownload>

in the directory for the clean database instance.

4. Start the newly created database instance by executing

```
virtuoso-t +service start +instance odcsclean
```

5. Copy `virtuoso.ini-dirty` to the directory for the dirty database instance created in step 1. Create a copy of the file and name it `virtuoso.ini`. If you are familiar with Virtuoso, you may customize configuration in `virtuoso.ini` according to your needs.
6. Create a new database instance by executing

```
virtuoso-t +service create +instance odcsdirty +configfile virtuoso.ini
```

in the directory for the dirty database instance.

7. Start the newly created database instance by executing

```
virtuoso-t +service start +instance odcsdirty
```

Security note. Don't forget to change credentials for your database instance.

3. Step by Step Installation Guide

ODCleanStore can be installed with a provided installer with graphical interface.¹

3.1 Prerequisites

Before installation, make sure your system satisfies requirements listed in Chapter 2. In particular, make sure to have two Virtuoso database instances created and *started* before installation, as described in Section 2.1.

3.2 Using the Installer

In order to run the installer, execute in the root directory of the distribution:

```
./odcs-installer.sh
```

on Linux, or run

```
odcs-installer.bat
```

on Windows. This should start the installer. If the Virtuoso's `isql` executable is not on system PATH for any reason, you must pass the full path to the installer manually:

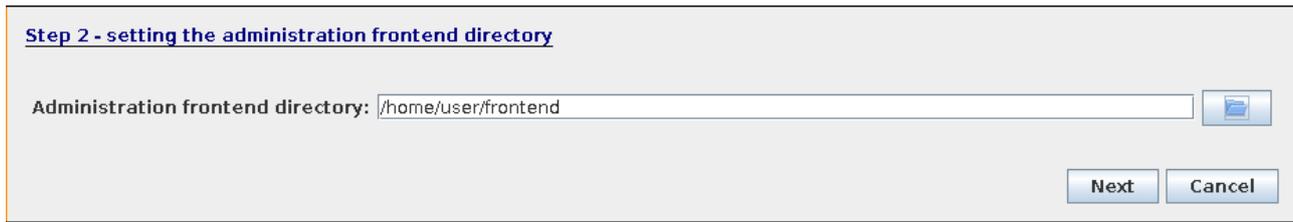
```
java -jar bin/odcs-installer-<version>.jar <full-path-to-Virtuoso-isql>
```

On the first screen of the installer, choose the directory where ODCleanStore binaries should be installed.

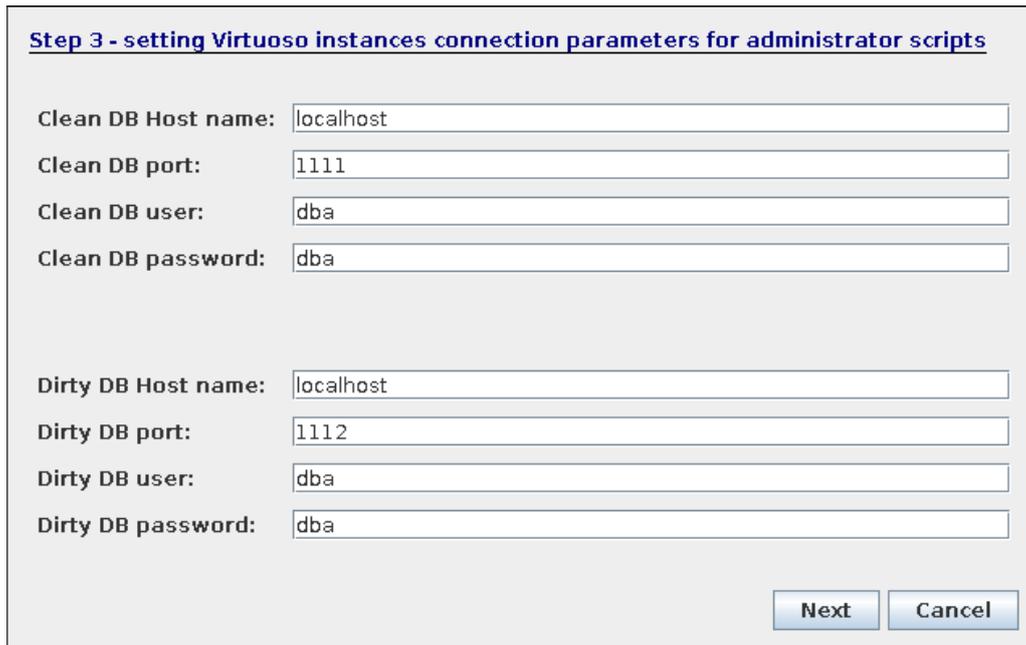


On the second screen, you can enter the directory where Administration Frontend web archive will be deployed. This should be a directory for web applications of your servlet container. In Tomcat, for example, it would be `$CATALINA_BASE/webapps`.

¹Alternatively, ODCleanStore can be deployed manually. Steps for the manual installation are described in the `README.txt` file in the distribution, the manual installation is intended for experienced users and developers, however, and is not officially supported.

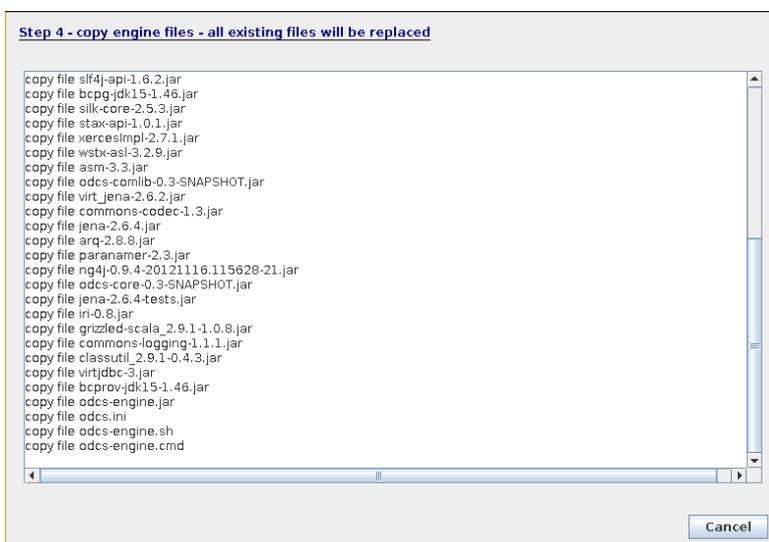


Next, enter the connection settings for the clean and dirty Virtuoso database instances. See Section 2.1 for more information about Virtuoso database instances.



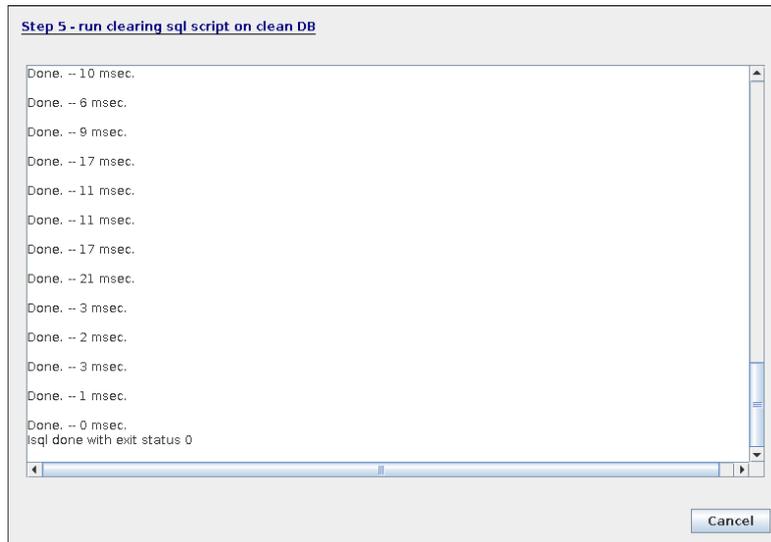
After entering valid connection settings, the installer starts the installation process. The installer will:

1. Copy Engine binaries and related files to the chosen Engine installation directory.



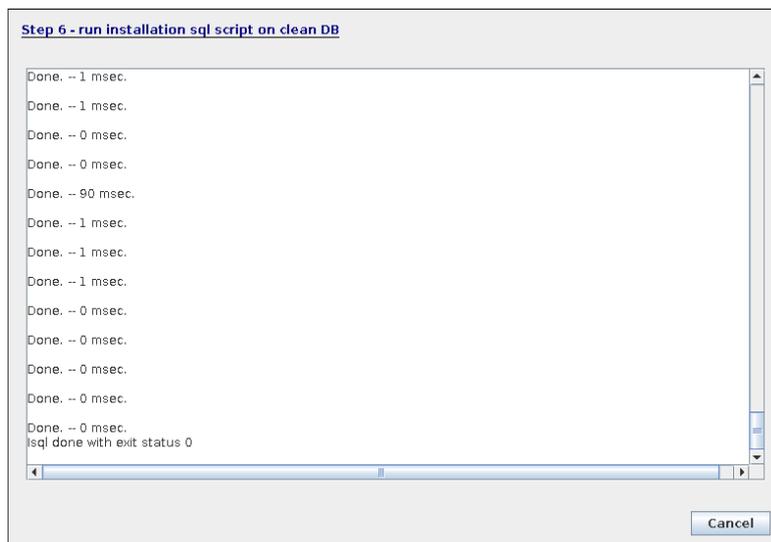
2. Copy the default configuration file to the Engine installation directory.
3. Initialize database instances:

- (a) Clear old tables in the clean database, if there are any.



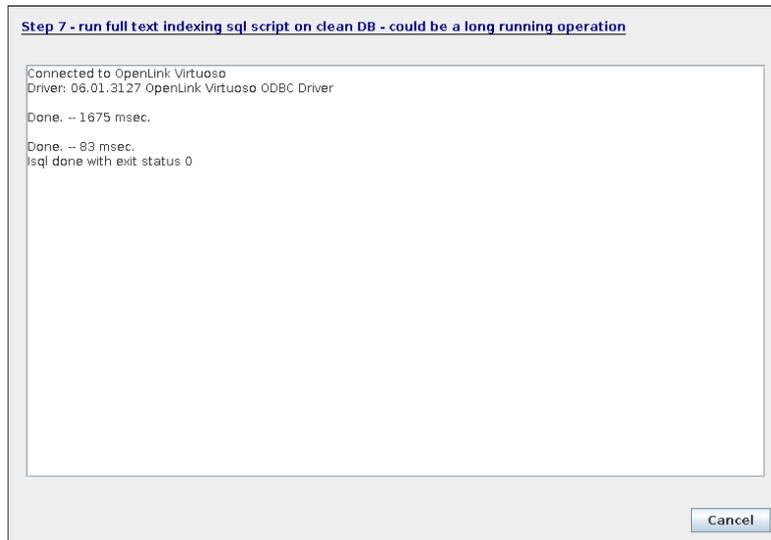
```
Step 5 - run clearing sql script on clean DB
Done. --10 msec.
Done. -- 6 msec.
Done. -- 9 msec.
Done. --17 msec.
Done. --11 msec.
Done. --11 msec.
Done. --17 msec.
Done. --21 msec.
Done. -- 3 msec.
Done. -- 2 msec.
Done. -- 3 msec.
Done. --1 msec.
Done. -- 0 msec.
Isql done with exit status 0
Cancel
```

- (b) Import new database tables in the clean database.



```
Step 6 - run installation sql script on clean DB
Done. --1 msec.
Done. --1 msec.
Done. -- 0 msec.
Done. -- 0 msec.
Done. -- 90 msec.
Done. --1 msec.
Done. --1 msec.
Done. --1 msec.
Done. --1 msec.
Done. -- 0 msec.
Isql done with exit status 0
Cancel
```

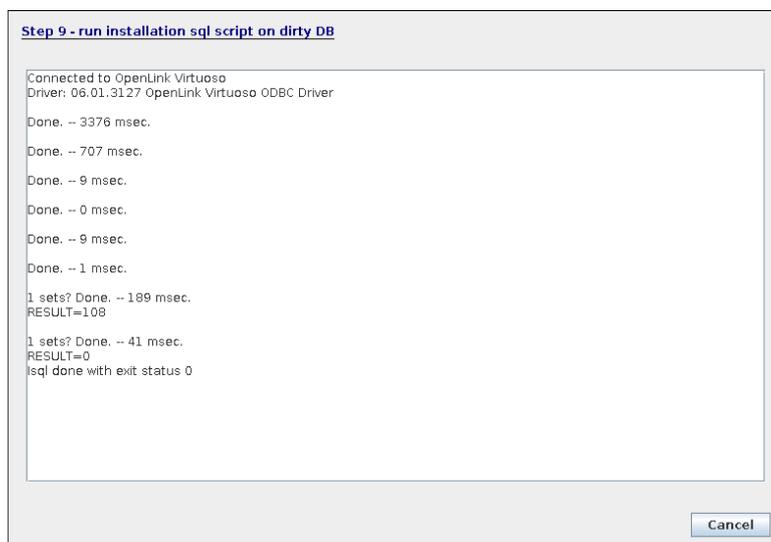
- (c) Create a fulltext index for RDF data in clean database (in order to enable keyword queries).



(d) Clear old tables in the dirty database, if there are any.



(e) Import new database tables in the dirty database.

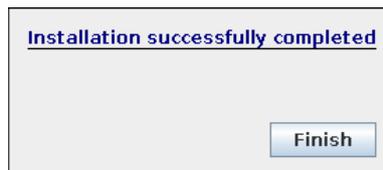


4. Adjust Administration Frontend web archive so that it contains a proper path to the de-

ployed configuration file and copy it to the chosen Administration Frontend installation directory.



If the installation was successful, the last screen of the installer is displayed, or an error message describing what went wrong is displayed otherwise.



3.3 Post-installation Steps

After successful install, you may customize default ODCleanStore settings in the configuration file `odcs.ini` in the Engine installation directory. See Chapter 6 for description of all options.

The installer deploys the web archive with Administration Frontend to the designated directory in your servlet container. Depending on the configuration of your servlet container, you may need to add the Administration Frontend web application to its configuration explicitly. If that is the case, in Tomcat, for example, this can be accomplished by adding a line similar to the following to `<Host>` section of Tomcat's `server.xml`:

```

<Context path="" docBase="{path to odcs-webfrontend-<version>.war}">
</Context>
  
```

After a clean installation, Administration Frontend contains two user accounts: an administrator account with username “adm”, password “adm”, and a user in the Data Producer (SCR) role with username “scraper”, password “reparcs”. For security reasons, **make sure to change the default credentials**. Also, the installer creates a user “SILK” with password “odcs” in the dirty database instance and grants it the `SPARQL_UPDATE` role. You should change the credentials for this user and update the affected `db.dirty_update.sparql.*` configuration options in `odcs.ini`.

4. Starting and Stopping the Application

4.1 Engine

ODCleanStore Engine can be started by executing the `run-engine.cmd` script (on Windows) or `run-engine.sh` script (on Unix) in the installation directory of ODCleanStore Engine. The Engine will load its configuration from the `odcs.ini` file in this directory and, after initialization, automatically start listening on Input and Output Webservice ports for requests.

In order to stop Engine, simply press `Ctrl+C` in the console window with Engine (send `SIGINT` signal to the Engine process on Linux). Engine will shutdown all services, wait (up until a timeout) until the currently active pipeline transformer finishes, and stop.

Engine is also prepared to be run as a system service on Windows, or daemon on Linux, respectively. Installing ODCleanStore as a service/daemon will be added to the installer in future.

4.2 Administration Frontend

Administration Frontend is a standard Java web application and its lifecycle depends on its servlet container – please refer to documentation of your server container¹. Usually, Administration Frontend would be reached by navigating to `http://localhost:8080/` in your web browser. Please note that Administration Frontend also needs the clean Virtuoso database instance to be running.

¹E.g. <http://tomcat.apache.org/tomcat-7.0-doc/index.html>

5. Trying out ODCleanStore

When Engine is started and Administration Frontend successfully deployed, you may try out ODCleanStore.

ODCleanStore can be managed from Administration Frontend in your web browser. Navigate to the address where it is deployed by your servlet container (e.g. `http://localhost:8080/`). By default, there is a single administrator account with username “adm” and password “adm”. In order to add new data processing pipelines in Administration Frontend, you will need to create a new user account, or add more roles to the default administrator account (log out and log in again for the changes to take effect).

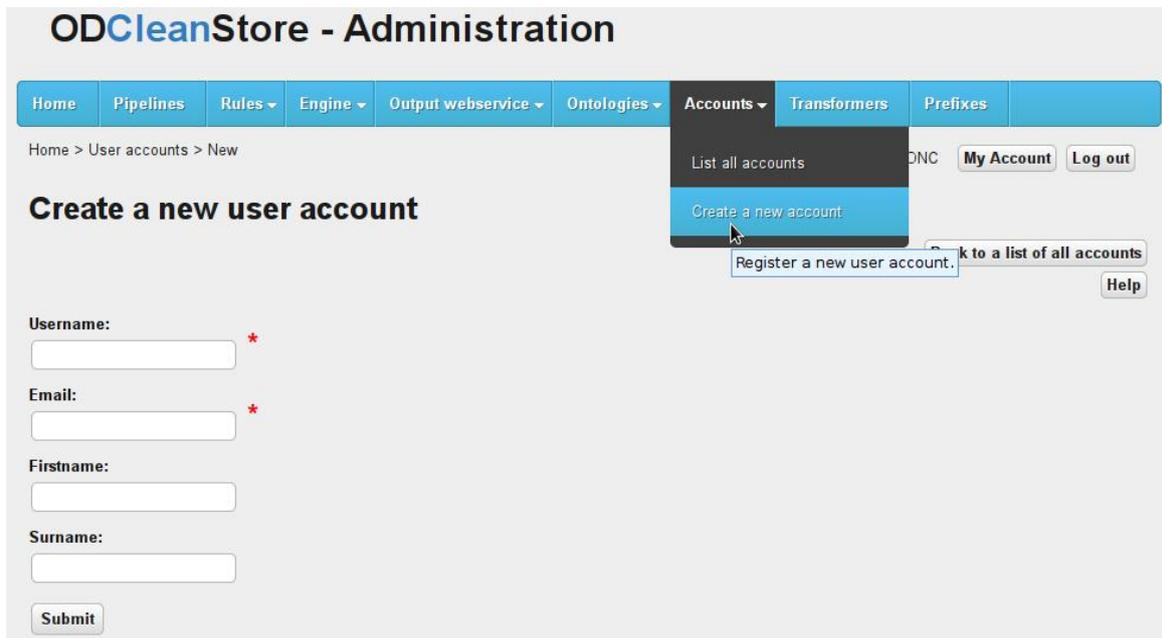


Figure 5.1: Administration Frontend– adding a new account

There should be one default pipeline created by default. You may customize it by assigning transformers to it. The functionality of Input and Output Webservices can be then demonstrated:

Sending data to ODCleanStore. To send data to Input Webservice, you can execute `run-example.cmd` or `run-example.sh` (on Windows or Linux, respectively) from directory `example/` in the distribution. The script will send data in `example-data.rdf` to Input Webservice together with metadata in `example-metadata.properties` and `example-provenance-metadata.rdf`

Parameters for Input Webservice can be changed in `example-metadata.properties`, most notably the name of the pipeline to process the data in the `pipelineName` option (the default pipeline will be used when none is specified) and location of Input Webservice.

Script `regenerate-uuid.cmd` (`regenerate-uuid.sh`) runs an utility that regenerates UUID in `example-metadata.properties`. This is necessary because every request to Input Webservice needs to have a unique UUID.

Querying over data. To query the stored data using Output Webservice, start with query-*.html documents in the example-queries/ subdirectory of the distribution. You can find more detailed information in User Manual.

URI query for <http://dbpedia.org/resource/Berlin>. Query executed in 1.569 s.

Subject	Predicate	Object	Quality	Source named graphs
dbpedia:Berlin	dbo:country	dbpedia:Germany	0.90000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia
dbpedia:Berlin	http://linkedgeodata.org/property/capital	"yes"	0.80000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/linkedgeodata
dbpedia:Berlin	rdfs:label	"Berlin"	0.94252	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/linkedgeodata, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/geonames, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/frebase
dbpedia:Berlin	frebase:location.geocode.longitude	"13.402740096987914" ^xsd:double	0.82446	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/linkedgeodata, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/geonames, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/frebase
dbpedia:Berlin	rdf:type	http://schema.org/City	0.92000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia, http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/frebase
dbpedia:Berlin	rdf:type	http://schema.org/Place	0.90000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia
dbpedia:Berlin	rdf:type	http://umbel.org/umbel/rc/Village	0.90000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia
dbpedia:Berlin	rdf:type	http://www.geonames.org/ontology#Feature	0.80000	http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/geonames

Source graphs:

Named graph	Data source	Inserted at	Graph score	License	Update tag
http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/dbpedia	http://dbpedia.org/page/Berlin	2012-04-01 12:34:56.0	0.9		
http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/error	http://example.com		0.8		
http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/frebase	http://www.freebase.com/view/en/berlin	2012-04-02 12:34:56.0	0.8		
http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/geonames	http://www.geonames.org/2950159/berlin.html	2012-04-03 12:34:56.0	0.8		
http://odcs.mff.cuni.cz/namedGraph/qe-test/berlin/linkedgeodata	http://linkedgeodata.org/page/node240109189	2012-04-04 12:34:56.0	0.8		
http://odcs.mff.cuni.cz/namedGraph/qe-test/germany/dbpedia	http://dbpedia.org/page/Germany	2012-04-05 12:34:56.0	0.9		

Figure 5.2: Sample results of an URI query

6. Configuration Options

Global configuration options for ODCleanStore are stored in configuration file `odcs.ini` in the Engine installation directory.

The configuration file is used both by ODCleanStore Engine and administration frontend. Configuration options are loaded only once when the Engine or administration frontend is started, therefore, it is necessary to restart the Engine or administration frontend web application for the changes to take effect.

List with a description of all configuration options follows:

Database Configuration

`db.clean.jdbc.connection_string`

Connection string for JDBC access to the Virtuoso instance representing the clean database. Must contain connection charset.

Default value: `jdbc:virtuoso://localhost:1111/CHARSET=UTF-8`

`db.clean.jdbc.username`

Username for JDBC access to the Virtuoso instance representing the clean database.

Default value: `dba`

`db.clean.jdbc.password`

Password for JDBC access to the Virtuoso instance representing the clean database.

Default value: `dba`

`db.clean.sparql.endpoint_url`

URI of the SPARQL endpoint for the clean database (used for reading only).

Default value: `http://localhost:8890/sparql`

`db.dirty.jdbc.connection_string`

Connection string for JDBC access to the Virtuoso instance representing the dirty database. Must contain connection charset.

Default value: `jdbc:virtuoso://localhost:1112/CHARSET=UTF-8`

`db.dirty.jdbc.username`

Username for JDBC access to the Virtuoso instance representing the dirty database.

Default value: `dba`

`db.dirty.jdbc.password`

Password for JDBC access to the Virtuoso instance representing the dirty database.

Default value: `dba`

`db.dirty.sparql.endpoint_url`

URI of the SPARQL endpoint for the clean database (used for reading only).

Default value: `http://localhost:8890/sparql`

`db.dirty_update.sparql.endpoint_url`

URI of the SPARQL endpoint which can be used for SPARUL modifications on the dirty database.

Default value: `http://localhost:8891/sparql`

`db.dirty_update.sparql.endpoint_username`

Username for the SPARQL endpoint given in `db.dirty_update.sparql.endpoint_url`.

Default value: `dba`

`db.dirty_update.sparql.endpoint_password`

Password for the SPARQL endpoint given in `db.dirty_update.sparql.endpoint_url`.

Default value: `dba`

Input Webservice Configuration

`input_ws.endpoint_url`

URL where the input webservice is listening.

Default value: `http://localhost:8088/inputws`

`input_ws.recovery_crash_penalty`

Waiting penalty after Input Webservice recovery crash before recovery restart in milliseconds.

Default value: `60000`

`input_ws.named_graphs_prefix`

Prefix of named graphs incoming where data & metadata are stored. Must be a valid URL.

Default value: `http://opendata.cz/infrastructure/odcleanstore/`

Output Webservice Configuration

`output_ws.result_data_prefix`

Prefix of named graphs and URIs where query results and metadata in the output are placed.

Default value: `http://opendata.cz/infrastructure/odcleanstore/query/`

`output_ws.port`

Port where Output Webservice runs.

Default value: `8087`

`output_ws.keyword_path`

Relative path fragment for the keyword query over the output webservice.

Default value: `keyword`

`output_ws.uri_path`

Relative path fragment for the uri query over the output webservice.

Default value: `uri`

`output_ws.metadata_path`

Relative path fragment for the metadata query over the output webservice.

Default value: metadata

`output_ws.named_graph_path`

Relative path fragment for the named graph query over the output webservice.

Default value: namedGraph

Query Execution Configuration

`query_execution.max_query_result_size`

Maximum number of results allowed in each database query performed during a query execution. This option effectively affects the maximum number of results returned in response to a query through Ouptut Webservice to a multiple of this number (cca 3 times the value of this option).

Default value: 500

Conflict Resolution Configuration

`conflict_resolution.agree_coefficient`

Agree coefficient used in aggregated quality formula – value N means that agreement of $N + 1$ sources with score 1 on the same value will increase the quality of that value to 1.

Default value: 4

`conflict_resolution.score_if_unknown`

Graph score used if none is given in the input of Conflict Resolution (e.g. is unknown).

Default value: 1

`conflict_resolution.named_graph_score_weight`

Weight of the named graph score in aggregated quality calculation (see `conflict_resolution.publisher_score_weight`)

Default value: 0.8

`conflict_resolution.publisher_score_weight`

Weight of the graph publisher's score in aggregated quality calculation (see `conflict_resolution.named_graph_score_weight`)

Default value: 0.2

`conflict_resolution.max_date_difference`

Difference between two dates when their distance is equal to MAX_DISTANCE constant, given in seconds. When the difference of two dates is higher or equal to this value, in Conflict Resolution, they are considered to be totally different; when the difference is smaller, the difference coefficient is proportionally smaller.

Default value: 31622400 (366 days)

Transformers Configuration

`object_identification.link_within_graph`

Sets whether to link incoming data against itself by default.

Default value: true

`object_identification.link_attached_graphs`

Sets whether to link data from attached graphs created by preceding transformers by default.

Default value: true

Engine & Backend Configuration

`backend.query_timeout`

Timeout for database queries executed by predefined transformers, Query Execution and Engine in seconds.

Default value: 30

`engine.clean_import_export_dir`

Directory for Clean Virtuoso instance data import and export files. Base for relative path is clean db server root (directory of the Virtuoso INI file). Path or his parent must be specified in the DirsAllowed param in the virtuoso INI file and the Virtuoso server restarted for access to be allowed to the files by Virtuoso.

Default value: odcs/

`engine.dirty_import_export_dir`

Directory for Dirty Virtuoso instance data import and export files. Base for relative path is clean db server root (directory of the Virtuoso INI file). Path or his parent must be specified in the DirsAllowed param in the virtuoso INI file and the Virtuoso server restarted for access to be allowed to the files by Virtuoso.

Default value: odcs/

`engine.startup_timeout`

Maximum time in milliseconds for services initializing before engine shutdown.

Default value: 30000

`engine.shutdown_timeout`

Maximum time in milliseconds for services shutdown before engine exit.

Default value: 30000

`engine.look_for_graph_interval`

Additional timer setting interval for lookup for graph for processing in milliseconds.

Default value: 8000

`engine.second_crash_penalty`

Waiting penalty after double pipeline crash before pipeline restart in milliseconds.

Default value: 60000

`engine.state_to_db_writing_interval`

Interval for writing engine state information to db in milliseconds.

Default value: 5000

`engine.engine_uuid`

Identifier of Engine instance when data processing is distributed over multiple parallel nodes. This setting has currently no effect and is reserved for future extensions.

Default value: 88888888-8888-8888-8888-888888888888

Administration Frontend Configuration

`web_frontend.gmail_address`

Username of a GMAIL account to send emails through.

Default value: odcleanstore@gmail.com

`web_frontend.gmail_password`

Password of a GMAIL account to send emails through.

Default value: odcleanstore2012

`web_frontend.output_ws_host`

Output webservice host.

Default value: http://localhost

`web_frontend.debug_directory_path`

Path to transformer directory for debugging.

Default value: ./odcs-debug

`web_frontend.output_ws_host`

Output Webservice host (used for references from Administration Frontend)

Default value: http://localhost

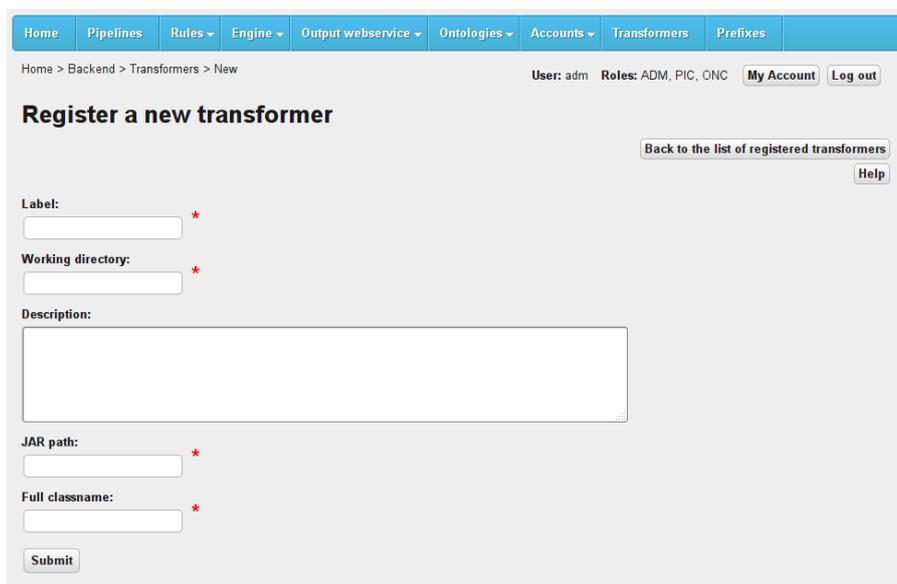
7. Custom Transformers

The administrator may extend data-processing capabilities of ODCleanStore by adding new custom transformers. In order to start using the transformer:

1. Implement your custom transformer by implementing the `Transformer` interface in Java or other compatible language.

The classfile for the `Transformer` interface is in `bin/odcs-core-<version>.jar` file in the ODCleanStore distribution, or can be obtained from sources in `odcs-core` maven artifact – see Chapter “Setting up Development Environment” in Programmer’s Guide. For more information about how to implement a custom transformer and what features it should have, please refer to chapter “Transformers – Introduction” in Programmer’s Guide.

2. Save the `.jar` file with the implementing classfiles to a location where Engine can read it from the filesystem. The classfiles of the `odcs-core` maven artifact need not be included in the `.jar` file (as they will be already loaded by Engine when the custom transformer is loaded at runtime).
3. In Administration Frontend, go to “Transformers” > “Add a new transformer” (Figure 7.1). Enter the path to the `.jar` file with the implementation, the fully qualified name of your class that implements the `Transformer` interface and fill in the other fields. Submit the form.
4. Now, the custom transformer is registered and can be assigned to pipelines and used for data processing.



The screenshot shows a web interface for registering a new transformer. At the top, there is a navigation bar with tabs for Home, Pipelines, Rules, Engine, Output webservice, Ontologies, Accounts, Transformers, and Prefixes. Below the navigation bar, the breadcrumb path is "Home > Backend > Transformers > New". The user information is "User: adm Roles: ADM, PIC, ONC" with buttons for "My Account" and "Log out". The main heading is "Register a new transformer". There is a "Back to the list of registered transformers" button and a "Help" button. The form contains the following fields:

- Label:** A text input field with a red asterisk.
- Working directory:** A text input field with a red asterisk.
- Description:** A large text area.
- JAR path:** A text input field with a red asterisk.
- Full classname:** A text input field with a red asterisk.

A "Submit" button is located at the bottom of the form.

Figure 7.1: Screen for registering a custom transformer

A. Distribution Contents

The distribution archive or CD contains ODCleanStore binaries, installer, documentation and some testing data. For a quick start, the relevant files for you are `README.txt` and scripts `install.cmd` or `install.sh` (for Windows or Linux, respectively) which start the installer.

Meaning of the other files and directories is explained below.

`bin/` Contains all Java binaries of ODCleanStore.

`engine/`

Contains binaries and dependencies for Engine part of ODCleanStore.

`webapp/`

Contains the Administration Frontend web archive.

`utils/`

Contains utilities. `regenerateUUID.jar` Generates a new UUID for metadata file used by `odcs-scraper`. `frontend-config.jar` is a utility for updating of path to the ODCleanStore configuration file in the Administration Frontend web archive.

`odcs-installer-<version>.jar`

Installer with a graphical user interface.

`odcs-scraper-<version>.jar`

A simple utility for testing of sending of data to Input Webservice.

`odcs-simpletransformer-<version>.jar`

A simple example of a custom transformer.

`odcs-core-<version>.jar`

`odcs-core` artifact. This can be used as a library when writing custom transformers, as it contains definition of the `Transformer` interface and other related classes.

`run-engine.cmd`, `run-engine.sh`

Scripts that start Engine for case of a manual installation.

`update-war-path.cmd`, `update-war-path.sh`

Scripts that update path to global configuration file in the Administration Frontend web archive. Supplied for convenience in case of manual installation.

`config/` Contains the default configuration file for ODCleanStore and recommended configuration files for Virtuoso database instances.

`odcs.ini`

The default global configuration file of ODCleanStore.

`virtuoso.ini-clean`

The recommended configuration file for the clean database instance of Virtuoso.

`virtuoso.ini-dirty`

The recommended configuration file for the dirty database instance of Virtuoso.

database/ Scripts for initialization of database instances.

install/

SQL scripts used by the installer.

clean_db/

SQL scripts for manual import for the clean database instance.

dirty_db/

SQL scripts for manual import for the dirty database instance.

doc/ Documentation for ODCleanStore.

javadoc/

API documentation of ODCleanStore sources in generated from javadoc.

LICENSE.txt

License under which ODCleanStore is distributed.

ODCleanStore User Manual.pdf

Documentation for ODCleanStore users.

ODCleanStore Administrator's and Installation Manual.pdf

This document.

ODCleanStore Programmer's Guide.pdf

Detailed documentation intended for developers.

sourceforge.txt

Link to the project homepage.

example/ Example data that can be send to Input Webservice.

example-data.rdf, example-data-isvzus.ttl

The actual payload data to be sent to Input Webservice.

example-metadata.properties, example-metadata-isvzus.properties

Property file with metadata sent together with payload data.

example-provenance-metadata.rdf

Example provenance metadata to be sent together with payload data.

run-example.cmd, run-example-isvzus.cmd

Script that sends the example data to Input Webservice.

regenerate-uuid.cmd

Script for updating UUID in metadata property files. This is necessary as every new input graph needs to have a unique UUID.

example-queries/ Simple forms in HTML that can be used for querying over Output Webservice. These forms are provided for testing and debugging purposes.

query-uri.html

Form for executing a URI query via Output Webservice.

`query-keyword.html`

Form for executing a keyword query via Output Webservice.

`query-named-graph.html`

Form for executing a named graph query via Output Webservice.

`query-metadata.html`

Form for executing a metadata query via Output Webservice.

`README.txt` File with basic information and instructions how to start with ODCleanStore.

`release-notes.txt` Release notes with information about changes between versions.

`install.cmd`, `install.sh` Scripts for starting the installer in Windows and Unix, respectively.

B. License

ODCleanStore is released as open source software under Apache License, Version 2.0.

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object

form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct

or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
 - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability

incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.