Semantic Web 0 (2016) 1–0
IOS Press

UnifiedViews: An ETL Tool for RDF Data Management

Tomáš Knap [a,*] Peter Hanečák [c] Jakub Klímek [a] Christian Mader [b] Martin Nečaský [a]
Bert Van Nuffelen [d] and Petr Škoda [a]

[a] *Charles University in Prague, Faculty of Mathematics and Physics*
Malostranské nám. 25, 118 00 Praha 1, Czech Republic
E-mail: knap@ksi.mff.cuni.cz
[b] *Semantic Web Company*
Mariahilfer Straße 70 / 8
A - 1070 Vienna, Austria
E-mail: c.mader@semantic-web.at
[c] *EEA s.r.o*
Hattalova 12B, 831 03 Bratislava, Slovak Republic
E-mail: peter.hanecak@eea.sk
[d] *TenForce bvba*
Havenkant 38, 3000 Leuven, Belgium
E-mail: bert.van.nuffelen@tenforce.com

Abstract. We present UnifiedViews, an Extract-Transform-Load (ETL) framework that allows users to define, execute, monitor, debug, schedule, and share data processing tasks, which may employ custom plugins (data processing units) created by users. UnifiedViews natively supports processing of RDF data. In this paper, we: (1) introduce UnifiedViews' basic concepts and features, (2) demonstrate the maturity of the tool by presenting exemplary projects where UnifiedViews is successfully deployed, and (3) outline research projects and future directions in which UnifiedViews is exploited. Based on our practical experience with the tool, we found that UnifiedViews simplifies the creation and maintenance of Linked Data publication processes.

Keywords: ETL, Data management, Data processing, RDF data, Linked Data

1. Introduction

The advent of Linked Data [1] accelerates the evolution of the Web into an exponentially growing information space where the unprecedented volume of structured data offers information consumers a level of information integration that has up to now not been possible. Data consumers can create mashups of Linked Data that leverage various data sources to support use cases which were not intended by the original data publishers.

Suppose a data wrangler wants to build an RDF data mart that integrates information from various RDF and non-RDF sources. So the wrangler's data processing task involves the activities of:

1. getting the data from certain data sources,
2. transforming the data to RDF data format,
3. cleaning the data,
4. interlinking it with other (external) data sources,
5. solving data conflicts to prepare the integrated data mart.

There are numerous tools used by the Linked Data community[1], which may support various phases of the data mart preparation; e.g., the wrangler may use *any23*[2] to extract non-RDF data and convert such data

*Corresponding author. E-mail: knap@ksi.mff.cuni.cz
*Corresponding author. E-mail: knap@ksi.mff.cuni.cz

[1] http://semanticweb.org/wiki/Tools
[2] https://any23.apache.org/

to RDF data format, *OpenLink Virtuoso*[3] database for storing RDF data and executing SPARQL (Update) queries [4,5], *Silk* [14] to interlink RDF data based on the declarative rules, or *LD-FusionTool* [9] to solve RDF data conflicts[4]. Nevertheless, using such tools directly, the data wrangler has to:

- configure every such tool differently,
- write a custom script downloading and unpacking data from various data sources,
- prepare a script executing a set of SPARQL Update queries curating the data,
- implement custom transformers, which, e.g., enrich processed data with the data from the DBpedia knowledge base [2], and
- write a custom script ensuring that the tools are executed in the required order, so that every tool has all the desired inputs when being launched.

Maintaining data processing tasks of increasing complexity is challenging. Suppose for example that the wrangler defines tens of such data processing tasks, which should run every month. So apart from the activities described above, the data wrangler has to also configure a scheduling script or use an external tool, such as *cron*[5], to ensure that the task is executed regularly. Furthermore, suppose that certain data processing task does not end as expected. To find the problem, the wrangler needs to query and browse through the intermediate results of the data processing task; this typically involves the need for manually setting up a triple store such as OpenLink Virtuoso and loading the suspicious intermediate RDF data data into it. Furthermore, if the data wrangler needs to review/adjust one of the data processing tasks later in the future, he has to re-examine the prepared scripts, recall the general idea of the task, the data flow between the tools etc.; in contrary, if there was a graphical visualization of the prepared task, which shows tools being used and the data flow between these tools, the data processing task would be documented and maintenance of such task would be much easier. Finally, the data wrangler cannot easily reuse configurations of the tools among the data processing tasks, so he cannot effectively reuse the already prepared tasks.

The task of compiling and setting up various tools of different vendors for multiple data analyses settings is cumbersome and often repetitive. In combination with the lack of an integrated debugging and maintenance support, the immediate consequence is a negative impact on a data wrangler's productivity. On a larger scale, we believe that the current lack of easy-to-use frameworks for Linked Data preparation and publication prevents many institutions to provide their datasets for public utilization as Linked Data.

Therefore, instead of requiring data wranglers to write most of the logic for defining, executing, monitoring, scheduling, and sharing the data processing tasks themselves, we provide UnifiedViews[6], an open-source Extract-Transform-Load (ETL) framework. It is an integrated solution that provides standard maintenance interfaces and lets data wranglers choose from various pre-defined and customizable "building blocks" to set up the individual data processing steps.

This paper is organized as follows. In Section 2, we present the basic concepts of UnifiedViews, how a data wrangler may interact with tool, and describe its architecture. To demonstrate the maturity of UnifiedViews, we introduce in Section 3 a number of exemplary projects in which the framework is successfully used. In Section 4, we summarize lessons learned from Section 3 and introduce current UnifiedViews research projects and future work. In Section 5 we provide an outline of related work in this problem domain and draw our conclusions in Section 6.

2. UnifiedViews Framework

An Extract-Transform-Load (ETL) *process* performs: 1) extraction of data from original data sources, 2) transformation of the data to the proper format and structure for the purposes of the consequent querying and/or data analysis 3) load of the resulting data into the target data source (also called operational data store, or data warehouse/data mart). An Extract-Transform-Load (ETL) *framework* allows users at least to define, monitor, and execute ETL processes.

UnifiedViews is an open-source ETL framework with a focus on processing RDF data. It allows users to define, execute, monitor, debug, schedule, and share Linked Data processing and publishing tasks. A sample task is the preparation of a data mart by the wrangler as introduced in Section 1.

A data processing task is represented in UnifiedViews as a *data processing pipeline* (or simply *pipeline*).

[3] http://virtuoso.openlinksw.com/
[4] https://github.com/mifeet/LD-FusionTool
[5] http://linux.die.net/man/8/cron
[6] http://unifiedviews.eu

Every pipeline consists of one or more *data processing units (DPUs)* and *data flows* between these DPUs.

Every DPU may declare certain mandatory or optional inputs, encapsulates certain business logic that processes the data (e.g., a DPU may extract data from a SPARQL endpoint, apply a SPARQL query, or transform CSV data to RDF data), and may produce certain outputs. DPUs may also provide a configuration dialog, so that a pipeline designer (e.g., the data wrangler mentioned above) may configure them differently in different pipelines. Administrators of the particular UnifiedViews installations may set up the default configurations of the DPUs and also prepare various alternative configurations, which may be directly used by pipeline designers.

A *data unit* is a container for data being consumed or produced by a DPU. We distinguish input and output data units. An input data unit contains data used as the input during a DPU's execution. An output data unit holds the data which is produced in the course of a DPU's execution. Every data flow between two DPUs X and Y consists of the output data unit of DPU X producing the data and the input data unit of DPU Y consuming the data. Every data unit has its name, which is assigned by a DPU developer and is mandatory. Data unit names are then visible to pipeline designers, so that pipeline designers may, e.g., map an output data unit of one DPU to input data unit of another DPUs. Every DPU may declare $n \in \mathbb{N}^0$ input data units and $n' \in \mathbb{N}^0$ output data units.

UnifiedViews supports three types of data units which can be both input and output data units and are distinguished by the type of information they can hold:

- *RDF* data units can contain RDF graphs,
- *Files* data units can contain files and folders, and
- *Relational* data units can contain tables from relational databases.

Every data unit can hold $m \in \mathbb{N}^0$ entries of the particular supported type.

There are four types of DPUs, which are determined by the number of input and output data units they declare, as well as their intended purpose:

- *Extractor*: A DPU that does not define any input data unit. Input data to such a DPU is not provided by the UnifiedViews framework, but rather obtained from external sources by the business logic of the DPU. For instance, an *extractor* may query data from a remote SPARQL endpoint or download files from a certain set of URLs.
- *Transformer*: A DPU that transforms inputs to outputs. It defines both input and output data units. UnifiedViews must ensure that proper inputs are prepared for the DPU and must also handle the outputs produced by the DPU. Examples of *transformers* are DPUs that transform tabular data to RDF data or execute SPARQL (Update) queries.
- *Loader*: A DPU that defines an input data unit, but does not define any output data unit. Output data produced by such a DPU is not maintained by the UnifiedViews framework, but rather intended for storage in external repositories outside of UnifiedViews. DPUs uploading data to a remote SPARQL endpoint or disseminating new records to the CKAN catalog[7] are examples of *loaders*.
- *Quality Assessor*: A DPU that assesses the quality of the input data and produces a quality assessment report as the output. We decided to distinguish these types of DPUs from transformers, because they work differently – they do not produce transformed data at the output, but rather produce quality assessment report. For instance, *quality assessor* DPUs may check to which extent the input data is complete or whether data type literals contain correct values in the resulting data.

UnifiedViews distinguishes *DPU templates* and *DPU instances*. The DPUs available in the system to be used on the pipeline by data wrangles are called DPU templates. Every DPU template defines a *template configuration* of the DPU. When a data wrangler places such DPU template on the pipeline, such placement is called *DPU instantiation* and the result is called a DPU instance. The DPU instance has its own *instance configuration* being based on the template configuration of the DPU. The DPU instance is always associated with the given pipeline. One pipeline may contain more different DPU instances of the same DPU template. Every DPU instance always points to the DPU template from which it was created.

2.1. Interacting with UnifiedViews

UnifiedViews provides a graphical user interface to define, manage, execute, monitor, debug, schedule, and share DPUs and pipelines. A screenshot of this interface is illustrated in Figure 1. It shows a data

[7] `http://ckan.org/`

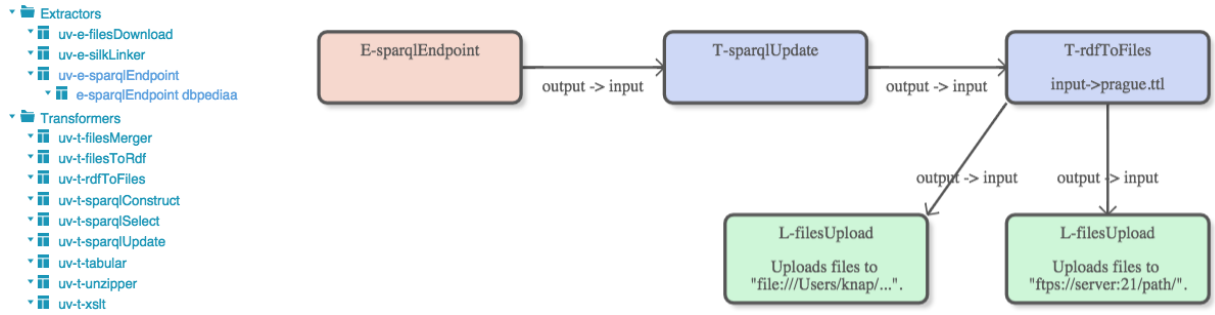

Fig. 1. UnifiedViews Framework – Definition of a Data Processing Task

processing pipeline, consisting of five DPUs (colored boxes) and four data flows (arrows connecting the boxes) between these DPUs. DPUs may be added to a pipeline by drag & dropping them on a pipeline canvas. Data flows between two DPUs may be denoted by drawing an edge between them. Labels on the data flow edges clarify which output data units are mapped to which input data units of the DPUs. In Figure 1, all data flows always map output data unit with the name *output* to input data unit of the next DPU with the name *input*; however, custom data unit names may be used by DPU developers, which are then reflected in the user interface.

As the pipeline is being prepared, UnifiedViews provides data wranglers with debugging capabilities; the data wrangler may execute a selected fragment of the pipeline at any time and browse or query (using the SPARQL query language) the entries in the input/output data units that are consumed/produced by each DPU.

When data wranglers are satisfied with the prepared pipelines, they can manually execute them and verify the results. Alternatively, it is possible to schedule pipelines for execution (1) once at certain time, (2) every certain period of time, or (3) after another pipeline is successfully executed. Data wranglers may also get notifications about the pipelines' execution states – either for all executions of the selected pipelines or only for those which ended with an error. It is also possible to get daily summaries about the executions of the selected pipelines in the last 24 hours.

UnifiedViews currently provides more than 35 *core DPUs*[8]. These are available for data wranglers in each deployment of UnifiedViews and provide basic functionality needed for

- obtaining external sources (CSV, DBF, XLS, XML files, RDF data, or relational tables),
- transforming them between various formats (e.g. CSV files to RDF data, relational tables to RDF data),
- executing typical transformations such as executing SPARQL Construct/Update queries, executing XSL transformations, linking/fusing RDF data, unpacking/packing/filtering files, and
- loading the transformed and curated data to external systems (loaders to various RDF databases and generic SPARQL endpoints, loaders of files via FTP/SFTP/etc., loaders to relational databases).

Apart from that, the UnifiedViews team[9] also provides additional DPUs. If a specific required functionality is not covered by a DPU available among the core DPUs or those provided by the UnifiedViews team, data wranglers may easily create and deploy their own custom DPUs. For this purpose, extensive documentation, such as tutorials, is provided online[10]. Basically, data wranglers may use UI wizards to generate skeletons for new DPUs and then they may use provided set of tutorials to declare input/output data units, consume inputs, produce outputs, generate events about the DPU's execution, add extensions to DPUs, etc.

All pipelines and DPU templates prepared by data wranglers are by default private (available only to the user who prepared them and also to administrators – see Section 2.2 for more details on roles). Nevertheless, UnifiedViews allows data wranglers to share prepared pipelines with other data wranglers, so that others can either see the pipeline (sharing in read-only mode) or even collaborate on the pipeline preparation (sharing with write access to the pipeline). Data wran-

[8] https://github.com/UnifiedViews/Plugins

[9] http://unifiedviews.eu/#get-team

[10] https://grips.semantic-web.at/pages/viewpage.action?pageId=50929588

glers may also create custom DPU templates (DPU templates available in the system with custom configurations) and share such DPU templates with others, so that others can use such custom DPU templates in their pipelines.

2.2. Architecture of UnifiedViews

UnifiedViews is composed of three main components:

- *Graphical user interface*: Being the primary means of interaction with the framework, the graphical user interface (henceforth referred to as *frontend*) supports definition, maintenance, execution, monitoring, debugging, scheduling, and sharing of data processing pipelines and maintenance of DPUs. The frontend is implemented in Java as a Web application using the Vaadin[11] framework. Figure 1 shows an excerpt of the screen where pipelines may be designed. Screencast of UnifiedViews user interface is available at `http://unifiedviews.eu`.
- *Pipeline execution engine*: It is responsible for running the (scheduled) pipelines, and implemented as stand-alone Java application (henceforth referred to as *backend*).
- *REST API administration service*: It allows to define, manage, execute, and monitor data processing pipelines without using the frontend. For example, external applications may execute pipelines and get results of the executions by interacting with this component.

Frontend and backend communicate via a relational database, which stores all configuration information such as pipeline setups, DPU configurations, execution states, or scheduled events.

To support scalability, multiple backend instances can run on different machines, effectively executing pipelines in parallel (see Figure 2). Each backend has its own identifier and all backends observe pending executions. If there is any execution pending, the first backend realizing that marks his identifier next to that pending execution and executes it.

Every backend uses its own *RDF Working Store* for storing temporary data which is produced by the pipeline during its execution. As the RDF Working Store, we currently support Sesame[12] repositories, either as a local native store or on a remote server. Experimental support for OpenLink Virtuoso is also in place and can be enabled by a configuration option. In general, any repository which supports the OpenRDF Sesame API (in the recent days renamed to Eclipse RDF4J API[13]), can be used as an RDF Working Store.

Every DPU is an OSGi[14] bundle. As a result, UnifiedViews can easily load/unload DPUs during runtime, allowing to, e.g., upgrade DPUs without restarting the framework. OSGi also ensures that two DPUs loaded to one UnifiedViews instance may use different versions of the same library without causing any conflicts.

UnifiedViews also supports authentication and authorization of users. Two roles are supported by default - *Users* and *Administrators*. Each such role can be associated with a list of permissions, e.g., a permission to import new DPU templates. Anytime a data wrangler wants to interact (view, edit, save, delete, etc.) with a certain entity (pipeline, DPU, scheduled event, etc.), UnifiedViews checks whether that user is authorized to do so. Spring Security[15] is used to ensure authentication and authorization of users.

There are two ways how DPUs on the pipeline may communicate certain information to a data wrangler when being executed. They may either publish an event (important message), e.g., that DPU *e-sparqlExtractor* was successfully executed and extracted 1000 triples, or just log something using standard mechanism for logging in Java. Both messages – events and logs – support various levels of severity (error, warning, info, etc.) and are displayed in the frontend of UnifiedViews as the pipeline is being executed. The reason why we distinguish events and logs (and also display them differently in the frontend) is to give data wranglers high-level overview of what happened during pipeline execution (through events) and to give them the possibility to examine logs in case more information is needed. DPUs may also throw an execution exception, which is semantically equivalent to sending an event with the severity *error*. When such exception is thrown, pipeline execution is stopped.

2.3. Availability of UnifiedViews

The source code of UnifiedViews is published under an open-source license which is a combination

[11] `http://vaadin.com`
[12] `http://rdf4j.org/`
[13] `http://rdf4j.org/`
[14] `https://www.osgi.org/`
[15] `http://projects.spring.io/spring-security/`

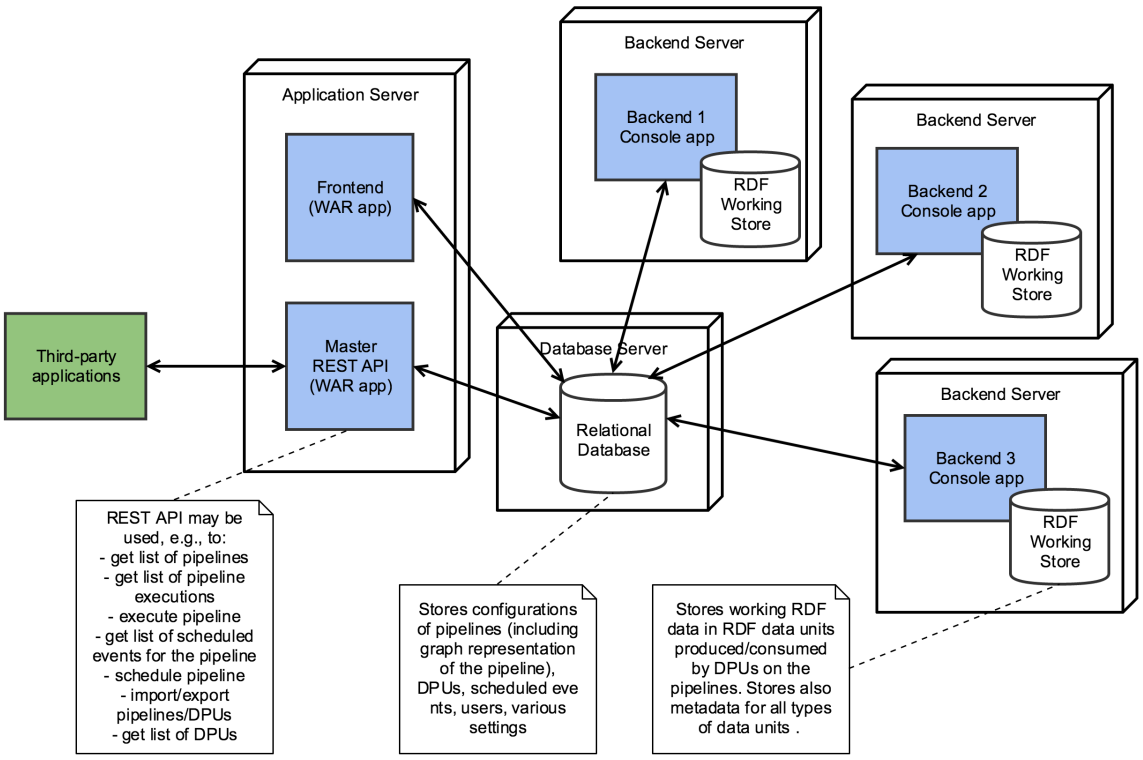

Fig. 2. UnifiedViews Framework – Architecture

of GPLv3[16] and LGPLv3[17] licenses. It is hosted at GitHub[18] and divided into the following repositories:

- *Plugin-devEnv*: It contains UnifiedViews APIs (for DPUs, configuration dialogs, data units, etc.) and also a set of helper classes, which simplify development of new DPUs.
- *Core*: It contains implementations of the UnifiedViews APIs from Plugin-devEnv, including also implementations of the supported data units, frontend, backend, and REST API administration service components.
- *Plugins*: It contains a set of core DPUs.
- *Plugins-QualityAssessment*: It contains DPUs which assess the quality of processed data.

All information about UnifiedViews, including the documentation and tutorial for building DPUs, may be found at the project's website[19]. For a quick start, UnifiedViews can be also easily deployed using the Docker[20] or Vagrant[21] images.

3. Deployments of the UnifiedViews Framework

In this section, we describe the projects in which UnifiedViews has been successfully deployed and used. For each project we describe in separate sections:

[16] http://www.gnu.org/licenses/gpl.txt
[17] http://www.gnu.org/licenses/lgpl.txt
[18] https://github.com/UnifiedViews

[19] http://unifiedviews.eu
[20] https://github.com/tenforce/docker-unified-views
[21] https://github.com/tenforce/vagrant-unifiedviews2.x-ubuntu15.04

– the motivation and goals of the project,
– approach we took and achievements we reached,
– challenges we faced and lessons learned.

Furthermore, Table 1 summarizes for each such project:

– the organization being responsible for realizing the project[22],
– number of UnifiedViews pipelines prepared in that project,
– information about the scheduling of the pipelines (whether they are scheduled to be executed every certain period of time or executed on demand),
– (approximate) number of RDF triples produced in each project (as of October 2016).

3.1. The Czech Trade Inspection Authority

The Czech Trade Inspection Authority in Czech Republic (CTIA)[23] examines and monitors fairness of businesses which supply or sell goods, provides services, or operates marketplaces in Czech Republic.

3.1.1. Motivation and Goals

Before CTIA published their core datasets – data about inspections, bans, and sanctions – as open data, lots of subjects (citizens, companies) requested access to particular aspects of the data (e.g., to get information about inspections and sanctions in the company X) based on the Czech act on free access to information[24] and CTIA always had to find resources to prepare and provide the requested data. So the goal of CTIA was to lower costs and publish all their core datasets upfront as open data to allow everyone to examine anytime any portion of the data, so that others can look up the needed information themselves.

3.1.2. Approach and Achievements

CTIA successfully used UnifiedViews to publish their core datasets about inspections, bans, and sanctions in CSV and RDF data formats according to recommendations of the OpenData.cz Initiative[25]. The datasets were available at their official web site[26]. Furthermore, two applications emerged interlinking the published CTIA data to other data sources (e.g., to other sources of inspections or to Business Register) and visualizing the data[27][28], which confirmed there was a public demand in CTIA data and it also directly showed the benefits of having data published as Linked Data.

3.1.3. Challenges and Lessons Learned

We find out that there are governmental institutions willing to publish their data as (Linked) open data and UnifiedViews was able to help them realize that goal.

Charles University, Department of Software Engineering, deploying UnifiedViews at CTIA, had to help CTIA with the initial installation/updates of UnifiedViews and with the pipeline design.

CTIA data wranglers had two issues when creating pipelines in UnifiedViews – (1) they were not Linked Data/RDF experts, thus, they did not know which ontologies they should use to publish their data as Linked Data in a correct and reusable way and (2) they sometimes did not know how certain DPUs should be interconnected in the pipelines to realize their particular need. Addressing issue (1) is more difficult and being able to semi-automatically suggest suitable RDF ontologies to represent source data is our future work; to address (2), we are working on the tutorials explaining how the DPUs should be interconnected in the typical data transformation and curation tasks.

Unfortunately, CTIA is currently (2016) not publishing new data in RDF data format, because employees involved in the preparation of the RDF data left CTIA and there was a political decision not to continue in that effort.

3.2. Council Open Data Initiative

The Council of the Europian Union (EU Council) is, together with the European Parliament, the legislative body of the EU. In 2015, the decision was taken to provide public information not only as documents but also as machine readable data to the EU citizens. The first dataset to be published were the Council's voting results. Later the Public Register (meta data on their documents) and the Requests to Access Documents were published.

[22] For details about the introduced organizations, such as their addresses/full names, please see affiliations of authors

[23] `http://www.coi.cz/en/`

[24] `http://www.zakonyprolidi.cz/cs/1999-106` (In Czech)

[25] `http://opendata.cz/en`

[26] `http://www.coi.cz/cz/spotrebitel/otevrena-data` (in Czech only)

[27] `http://www.spinque.com/czech-restaurant-inspections`, prepared by the Dutch company Spinque, `http://www.spinque.com/`

[28] `http://vysledkykontrol.cz/`, prepared by the OpenData.cz initiative

Table 1
Overview of the projects

Project	Realized by	Pipelines	Scheduled	Number of triples
The Czech Trade Inspection Authority	Charles University	5	on demand	~3.5 Millions
Council Open Data	Semantic Web Company/TenForce	14	bi-hourly/daily	~25 Millions
Open Data Support	TenForce	18	weekly	~45 Millions
Westtoer Datahub	TenForce	15	hourly/daily	~22 Millions
Slovak Environmental Agency	EEA	5	on demand	196 330
OpenData.cz Initiative	Charles University	311	varies	~557 Millions

3.2.1. Motivation and Goals

Until 2015, the votes of the EU Council were only available in an unstructured format as a picture embedded in a PDF document. Since voting is a core element of democratic accountability, there is a considerable interest among practitioners and researchers in the voting patterns at EU level, including those of the EU Council.

The goal of the project is to ensure transparency on information about the votings of EU Council, and to empower experts, journalists and citizens to re-use the data and analyse such votings as well as realize visualizations, applications, etc., on top of the EU Council dataset. The EU Council vote dataset does not only contain the votes but also information about, e.g., the act type (regulation, directive, decision or position), act number (as published in the EU's Official Journal), document number (submitted to the Council for adoption), inter-institutional number and much more.

The positive experience on the voting results dataset let to the second phase of that project during 2016 with the goal to provide the *Public Register* and the *Requests for Access to Documents* datasets as Linked Data. The Public Register is a meta data catalog of all documents that are publicly accessible. The Requests for Access to Documents provides insights in the requests the EU Council is receiving from the public to get access to documents related to certain topic. For instance, one can ask what are the documents related to the CETA treaty[29]. The EU Council responds with a list of document references, indicating their accessibility. As a side-effect of such requests, documents indicated as confidential might be opened up. Furthermore, the Requests for Access to Documents dataset provides anonymized logged requests and reponses; as a result, one can discover, e.g., the rate of opening documents to the public.

3.2.2. Approach and Achievements

From a technology perspective, the Council Open Data Initiative implements a mechanisms to extract data from the EU Council's original database. Making use of UnifiedViews, this data is then automatically converted into RDF data format by adoption of the Data Cube vocabulary and published using OpenLink Virtuoso RDF store. To achieve this, Semantic Web Company, realizing this project for EU Council, developed a pipeline which is scheduled and bi-hourly executed.

During the second phase, the Linked Data publishing environment has been made more robust and upgraded to the latest version of UnifiedViews by TenForce.

As a first example of date re-use, three data visualizations have been created: a map visualization[30], a visualization of votes over time[31] and votes on a punchcard[32]. To ensure currency, these visualizations are always created from the data directly taken from the Council Open Data Initiative's API.

3.2.3. Challenges and Lessons Learned

Due to internal policies, the project requires the use of MS SQL server instead of open-source databases like MySQL or PostgreSQL which were so far supported by UnfiedViews. We therefore adapted UnifiedViews to also operate on a MS SQL server for storing its internal data, i.e., pipeline stages, scheduling information or user management.

The second phase was confronted with a higher volume of data (about 3 millions documents) and the information had to be retrieved from an Oracle database. It turned out that getting access to the data was easy, the default extraction DPU of Unified-

[29] http://ec.europa.eu/trade/policy/in-focus/ceta/

[30] https://www.semantic-web.at/council/map

[31] https://www.semantic-web.at/council/votes_over_time/

[32] https://www.semantic-web.at/council/punchcard/

Views handled that case. The challenge was, however, throughput performance. It turned out that simply turning the oracle database views in RDF data (without any post-processing) was sufficiently performant but adding further transformations caused the throughput performance to drop significantly. An in depth analysis showed that the throughput performance is determined by the used RDF working store in UnifiedViews. The default Sesame working store, is very inefficient when dealing with graph management operations and deletion of triples. When using the Sesame working store, the size of the pipelines (the number of DPU's) combined with the starting volume is a measure of the throughput performance. By batching the input and splitting the pipelines in smaller ones, a situation has been established in which each pipeline execution requires at most 30 minutes.

A main lesson learned is the importance of high quality source data. This includes both the enforcement of strict syntax validation for all data elements, as well as an increased focus on using controlled vocabularies wherever applicable. Timeliness and up-to-dateness of the data is therefore crucial for that use case which can be achieved by UnifiedView's advanced scheduling functionality for the data extraction pipeline.

Another major insight gained from the project was, that we were able to lower the barrier of starting to work with Linked Data extraction and conversion methods. For the Voting Results dataset, Semantic Web Company provided both the technology as the transformation pipelines for achieving the project's goals. Only by using email interaction, Semantic Web Company was able to instruct the council members to maintain (i.e., control and observe). Given that prior to this project, the EU council had only minor experience with Linked Data, this can be considered a major achievement. During the second phase, this earlier experience paid off and it established a working environment in which the development and the later handover of the improved EU Council linked data processing setup went smoothly. As we speak today, the EU Council is working with internal resources to publish new datasets using UnifiedViews.

We can therefore conclude that UnifiedViews, as a user-friendly graphical tool, can encourage institutions to publish their data in an open format on the Web, contributing to increased availability of high quality Linked Data.

3.3. Open Data Support: First pan-European Open Data Portal

The European Commission Directorate General for Communications Networks, Content & Technology (DG Connect)[33] is fostering the uptake and knowledge on Open Data within the European Union and its 28 member states. This is done by a multitude of actions ranging from funding research and innovation, market studies, to projects within the European Commission. Notable outcomes are the European Union Open Data Portal[34] and the European Open Data Forum[35].

3.3.1. Motivation and Goals.

DG Connect launched a 3-year project Open Data Support[36] to make governmental data throughout the European Union more accessible (2013-2015). The ambition was to increase the awareness on (Linked) Open Data in all member state administrations and to improve the visibility and facilitate the access to datasets published on national open data portals in order to increase their re-use within and across borders.

3.3.2. Approach and Achievements.

The awareness on (Linked) Open Data has increased as the project has trained over 1200 persons active in governmental administration in almost every member state.

The second ambition initiated the creation of the DCAT-AP specification[37]. DCAT-AP harmonizes and adopts the W3C DCAT specification[38] to the European context, creating so a more uniform dataset description within the EU. The specification has been adopted by all member states as the metadata vocabulary for Open Data catalogues. Using DCAT-AP, the project realized the first pan-European Open Data Portal containing dataset descriptions aggregated from 18 national open data portals of the member states. It resulted in a collection of more than 80000 datasets; for this collection process, TenForce decided to use UnifiedViews to harvest, compare, and harmonize open

[33] https://ec.europa.eu/dgs/connect/en/content/dg-connect

[34] http://open-data.europa.eu

[35] http://www.data-forum.eu/

[36] https://joinup.ec.europa.eu/community/ods/description

[37] https://joinup.ec.europa.eu/asset/dcat_application_profile/asset_release/dcat-ap-v11

[38] http://www.w3.org/TR/vocab-dcat/

data. The Open Data Support harvesting pipeline has been the first commercial application of UnifiedViews.

3.3.3. Challenges and Lessons Learned.

Despite being the first commercial application of UnifiedViews, the benefits of the UnifiedViews approach for this challenge were immediately visible. When having the first DPU's ready for this task (i.e., extractors from CKAN catalog, loaders to CKAN catalog), the actual aggregation and harmonisation work could have started. It allowed to reach within a short amount of time (less than 2 months) a first production ready setup. Thereafter an iterative approach could have been followed to stepwize improve the quality of the already harmonised datasets and to add more new functionality such as: versioning, automated translation of descriptions and titles and DCAT-AP compatibility correspondence (quality report). At the same time a methodology for harvesting any new data portal was established, which reduced the inclusion of a new data portal from (initially) 10 days to 2 days.

For this project the UnifiedViews RDF working store was switched from Sesame to OpenLink Virtuoso. This was done to achieve a better throughput performance. Despite this choice realized the needed performance improvement, it created another challenge: the proper support of SPARQL query language. It turned out that Virtuoso is not supporting the same SPARQL queries as Sesame, therefore SPARQL queries had to be rewritten to satisfy the restrictions imposed by OpenLink Virtuoso.

During the 3 years of the project the maintenance of the pipelines has been done by several persons. The transition from one to another was each time rather smooth, the only prior knowledge of each maintainer was general Linked Data experience. This indicates that UnifiedViews also assists in the knowledge transfer that is required in any long lasting project.

3.4. Westtoer Datahub

Westtoer[39] is a Flemish governmental agency supporting the touristic actors (cities, organisations, etc.) active in the coastal area of Belgium. Their main role is the coordination between these actors, but they also participate in activities such as marketing, development of new touristic initiatives and the management of cycling and hiking routes.

3.4.1. Motivation and Goals.

In the context of Westtoer's role as a knowledge center of the touristic information, touristic data is being collected and made available. In the recent past Westtoer established a datahub: a data portal from which machine processable data is made available[40]. The datahub is a service Westtoer offers to its partners: instead of each individual application / software solution collecting the touristic data from all data suppliers themselves, Westtoer will provide the data via a centralised datahub. Besides that the touristic actors only have to discuss with a single data provider, they also benefit from the data integration work being done by the Westtoer datahub; for instance, the knowledge on how accessible a location is can be combined with the events that take place there.

3.4.2. Approach and Achievements.

For the Westtoer datahub UnifiedViews is deployed as a dockerized solution exporting the data to an OpenLink Virtuoso RDF store. End-users can access the data via the DataTank[41].

At Westtoer, a locally developed tool was taking care of the data conversion. The drawbacks of that tool: limited maintainability due to complex specifications, UTF-8 encoding problems, etc., and the need for transforming (new) data sources to a new vocabulary encouraged the team to replace it with UnifiedViews.

After an extensive test period, the setup has been validated by the active data consumers (there are running products on the datahub). As a result, the new setup with Unifiedviews will be taken into production at the end of 2016.

3.4.3. Challenges and Lessons Learned.

As experienced in other use cases, the creation of the UnifiedViews pipelines is a labor intense work requiring knowledge of source and target vocabularies. The large domain and the wide variety of sources made that each source required extra deep assessment. Furthermore the target domain is open ended. So during the pipeline creation the understanding grows by inspecting and evaluating the result with the end-users. As this is a human process, it showed a slow progress.

The initial pipelines, which are based entirely on SPARQL (Update) queries, reached not always the required throughput performance. Applying the knowledge from other projects was able to create some improvements, but still it was not sufficient especially for

[39] http://www.westtoer.be

[40] http://datahub.westtoer.be

[41] http://thedatatank.com/

one source which was provided as a set of XML documents. The solution was to rewrite the original XSLT that turned the input into basic RDF data. Further reduction has been reached by incorporating some domain knowledge. In the end, the execution went from 100 hours to only 45 minutes.

The Westtoer DataHub UnifiedViews setup is now in its first release. The current experience allowed to identify specific improvement actions. The knowledge that those actions can be implemented without interfering the whole setup, but only that part that must be addressed, creates comfort for the maintainer.

3.5. Slovak Environmental Agency

Slovak Environmental Agency (SAE) is the provider of the data from the environmental domain; this includes data about environmental burdens, protected sites, land cover, waste dumps etc. SAE is also an infrastructure provider – it hosts DB servers and web services working with the environmental data.

3.5.1. Motivation and Goals.

SEA wanted to explore the potential to increase reuse of their data if published as Linked Data. SAE decided to publish as Linked Data datasets on: protected sites, species distribution, bio-geographical regions, land cover, contaminated sites registered as enviromental burdens; these datasets are available in the Geography Markup Language (GML) via an API provided by the Web Feature Service, typically in INSPIRE format[42]. So the goal was to harvest the selected datasets, convert them to RDF data format, interlink them with relevant Linked Data resources, provide visualizations and interface for querying the published data.

3.5.2. Approach and Achievements.

UnifiedViews was successfully deployed (as one of components of Open Data Node (ODN) publication platform[43]) on the remote cloud infrastructure of SEA. During pilot collaboration with COMSODE project[44], EEA company built a data transformation pipeline in UnifiedViews, which harvested the data from the SAE's data service, converted it to RDF via XSL transformations and enriched the datasets with links to external datasets including GeoNames[45] and datasets from the European Environmental Agency: Biogeographical regions 2011, Natura 2000 and EUNIS[46]. The results of the transformations were published using other components of ODN: UnifiedViews loaders, CKAN catalog and a Linked Data Visualization Model implementation [6]. The catalog with published data is available online[47].

3.5.3. Challenges and Lessons Learned.

Initial barrier we had to overcome was that the vocabularies mapping the INSPIRE XML schemas to RDF were not available, so we had to provide the mappings.

UnifiedViews was able to transform, enrich and publish RDF data in a simple way, allowing easy maintenance for the future. A key benefit of the RDF version of the SEA datasets is that it is straightforward to combine them with third-party datasets.

3.6. OpenData.cz Initiative

OpenData.cz initiative[48] is the initiative of a group of people mainly from Charles University in Prague and University of Economics, Prague.

3.6.1. Motivation and Goals.

The goal of the initiative is to extract, transform and publish Czech open data in the form of Linked Data, so that the initiative contributes to the Czech Linked Open Data cloud. The initiative focuses mainly on Czech governmental data.

3.6.2. Approach and Achievements.

For this effort, UnifiedViews framework has been successfully used since September 2013; so far OpenData.cz initiative has published over 70 datasets and hundreds of milions of triples. The list of published datasets is available online[49].

3.6.3. Challenges and Lessons Learned.

Using UnifiedViews to maintain data processing tasks of OpenData.cz initiative simplified the data processing tasks and at the same time kept the data processing tasks documented.

OpenData.cz initiative realized that certain fragments of pipelines tend to repeat for many pipelines,

[42] http://inspire.ec.europa.eu/index.cfm/pageid/2/list/datamodels
[43] http://opendatanode.org/
[44] http://www.comsode.eu/
[45] http://www.geonames.org/
[46] http://www.eea.europa.eu/data-and-maps
[47] http://data.sazp.sk/
[48] http://opendata.cz
[49] http://linked.opendata.cz/en

e.g., the pipeline fragment producing DCAT-AP metadata for the published data; to simplify creation of new and maintenance of existing pipelines, UnifiedViews should have the possibility to pack these fragments, so that they may be placed on the pipelines in the same way as other DPUs.

Further, although it was really effective to manage pipelines in UnifiedViews, sometimes it happened that scheduled pipeline suddenly did not produce results as expected; the reasons for that were typically twofold – either the structure of the source data changed in the meanwhile or the pipeline designer made a mistake as he was fine-tuning the pipeline definition. In these cases, UnifiedViews should send alerts to the pipeline designer that the results of the scheduled pipeline suddenly changed dramatically.

4. Summary of Lessons Learned and Future Work

Most of the projects in Section 3 confirmed that UnifiedViews provides easy pipeline management via user interfaces. *Open Data Support* project also mentioned that UnifiedViews assists in the knowledge transfer (between data wranglers) that is required in any long lasting project.

Predefined UnifiedViews plugins which may be used out of the box and without a need for heavy programming speed-up the preparation of data processing tasks; *Open Data Support* project explicitly quantified that time needed to include yet another data source to be extracted was reduced from 10 to 2 days when UnifiedViews was used.

Westtoer Datahub and *Open Data Support* mentioned performance issues when processing bigger amounts of data with UnifiedViews – these issues are planned to be solved by adding solid support for Openlink Virtuoso as the RDF working store and by optimizing the way how RDF data is passed between DPUs.

OpenData.cz initiative project realized that certain fragments of pipelines tend to repeat often; to simplify creation of new and maintenance of existing pipelines, UnifiedViews should have the possibility to pack these fragments, so that they may be placed on the pipelines in the same way as other DPUs. They claimed that UnifiedViews should send alerts to the pipeline designers when something suspicious happened during pipeline execution, e.g., when the results of the scheduled pipeline suddenly changes dramatically. We plan to provide a possibility for DPU developers to include to their DPUs RDF validation features via a single line of code; as a result, for DPUs supporting the RDF validation features data wranglers may define sets of (SPARQL ASK) queries verifying the produced RDF data.

As reported by *Westtoer Datahub* and *The Czech Trade Inspection Authority*, data wranglers had difficulties when preparing data processing pipelines, because 1) they are not Linked Data experts and do not know source and target vocabularies they should use and 2) they sometimes did not know how certain DPUs should be interconnected in the pipelines to realize their particular need. To address (1) as a future work we plan to include algorithms for (semi)automatic suggestions of suitable RDF ontologites representing source data.

4.1. UnifiedViews in Research Projects

Currently, we also plan to use UnifiedViews in several research projects.

The goal of the EU-funded ALIGNED project[50] is to better integrate the software and data development lifecycles. In that context, UnifiedViews will be used to extract data from enterprise knowledge management and issue tracking systems such as Atlassian Confluence and Jira. An additional goal of ALIGNED is to provide approaches and tools for detection and curation of consistency violations in Linked Data. As UnifiedViews can be configured to check datasets periodically or on certain events, it is the right tool for supporting the ALIGNED project's requirements. In the course of the project, Semantic Web Company therefore plans to contribute additional DPUs that implement the checks and data curation algorithms that are developed during the ALIGNED project.

ADEQUATe, a FFG-funded[51] project where Semantic Web Company is involved in, focuses on researching effective methods for automatic conversion of existing public data into (Linked) Open Data, which involves various data cleansing methodologies that can be reused also in other projects and use cases. Based on the experience we have gained with the deployments described above, we believe that UnifiedViews will prove as an effective tool for helping to accomplish the goals of the ADEQUATe project.

[50] http://aligned-project.eu/

[51] Austrian Research Promotion Agency (FFG), https://www.ffg.at/en

Within the EU-funded YourDataStories[52] (YDS) project, UnifiedViews will be used as the key data transformation tool. In YDS, economic data from Ireland and Greece and development aid information from the Netherlands are converted and aligned to the common data model. TenForce is responsible for training the YDS partners and TenForce's knowledge engineers to use UnifiedViews for data transformation tasks in YDS. As was confirmed by the lessons learned from the projects in Section 3, data wranglers may need further assistance when developing pipelines in UnifiedViews, e.g., with the proper mappings of source data to target ontologies; in YDS we will try to address these shortcomings.

5. Related Work

There are plenty of ETL frameworks for preparing tabular data to be loaded to data warehouses, some of them are also open-source[53] – for example Clover ETL (community edition)[54], KETL[55], or EplSite ETL[56]. There is also GeoKettle[57], which focuses on integration of different spatial data sources for building and updating geospatial data warehouses. In all these frameworks custom DPUs may be created in some way, but the disadvantage of these non-RDF ETL frameworks is that there is no support for RDF data format and ontologies in the frameworks themselves. As a result, these non-RDF ETL frameworks do not have direct support for exchanging RDF data among DPUs, they are not prepared to suggest ontological terms in DPU configurations, a feature important when preparing SPARQL queries or mappings of the table columns to RDF predicates, etc. Furthermore, there are typically no DPUs able to extract RDF data from external SPARQL endpoints, transform RDF data in the working RDF store, convert RDF data from/to other data formats, such as CSV/Excel files, or load RDF data to external SPARQL endpoints. Hence, as RDF support is the outstanding characteristic of UnifiedViews, we focus on the area of RDF ETL frameworks in the remainder of this section.

ODCleanStore[8] is a Java based Linked Data Management framework developed at Charles University in Prague, Department of Software Engineering. Linked Data Manager (LDM)[58] is a Java based Linked (Open) Data Management suite to schedule and monitor required ETL tasks for web-based Linked Open Data portals and data integration scenarios. LDM was developed by Semantic Web Company[59]. As part of LOD2 project[60], ideas from LDM were integrated into ODCleanStore, which was then renamed to UnifiedViews. In particular, it improves: (1) the robustness of the pipeline execution engine, (2) usability of the graphical user interface, and (3) simplicity of new DPUs' creation.

DERI Pipes [10] is an engine and graphical environment for general Web data transformations. DERI Pipes supports creation of custom DPUs; however, an adjustment of the core is needed everytime a new DPU should be added; in UnifiedViews, it is possible to reload DPUs as the framework is running. DERI Pipes also does not provide any solution for library version clashes; on the other hand, in UnifiedViews, DPUs are loaded as OSGi bundles, thus, it is possible to use two DPUs requiring two different versions of the same dependency (library) and no clashes arise. In DERI pipes, it is not possible to debug inputs and outputs of DPUs. Lastly, DERI pipes seems to be unmaintained for years.

Linked Data Integration Framework (LDIF) [12] is an open-source Linked Data integration framework that can be used to transform Web data. The framework consists of a predefined set of DPUs, which may be influenced by their configuration; however, new DPUs cannot be easily added[61]. LDIF provides a user interface to monitor results of executed tasks; however, when compared with UnifiedViews, LDIF does not provide any graphical user interface for defining and scheduling tasks, managing DPUs, browsing and querying inputs to and output from the DPUs, and managing users and their roles in the framework. LDIF also does not provide any possibility to share pipelines/DPUs among users. On the other hand, LDIF provides possibility to run tasks using Hadoop[62].

[52] http://yourdatastories.eu/

[53] http://sourceforge.net/directory/business-enterprise/enterprise/data-warehousing/etl/

[54] http://www.cloveretl.com/products/community-edition

[55] http://sourceforge.net/projects/ketl

[56] http://sourceforge.net/projects/eplsiteetl

[57] http://www.spatialytics.org/projects/geokettle/

[58] https://github.com/lodms/lodms-core

[59] http://www.semantic-web.at

[60] http://stack.lod2.eu/blog/

[61] http://ldif.wbsg.de/

[62] http://hadoop.apache.org/

Grafter[63] allows specification of the transformation pipelines that convert tabular data into either more tabular data or Linked Data graphs. Grafter is targeted at software developers, Clojure[64] language is used to prepare pipeline definitions. DataGraft[65] is a graphical user interface for defininig pipeline definitions – how tabular data (CSV/Excel files) should be converted to tabular or Linked Data. Preparation of pipeline in DataGraft is again intended for developers, who are able to write Clojure functions. When comparing UnifiedViews and Grafter/DataGraft, the latter one supports only transforming tabular data (CSV/Excel) to Linked Data; on the other hand, UnifiedViews provides 35+ core DPUs for wider range of transformations (executing SPARQL queries, XSL transformation, working with relational data, executing Silk linker, etc.). Further, DataGraft does not allow easy creation of new plugins – it is a one purpose tool to support conversion of tabular data to Linked Data; on the other hand, UnifiedViews provides an easy and documented way to prepare new DPUs. DataGraft provides a possibility to host the transformed data in the cloud-based triplestore, share data privately or publicly, and publish data in a catalog; such functionality is not available in UnifiedViews, nevertheless it is covered by Open Data Node[66]. Grafter is also not that matured as UnifiedViews, it is under active development and authors plan breaking changes in the next future.

Booth [3] presents an approach to automate data production pipelines using semantic web technologies. Every pipeline consists of nodes composed of two parts: the *updater* and a *wrapper*. A wrapper is a standard component that is responsible for invoking the updater, communicating with other nodes, caching results; an updater executes the business logic of the node. The approach is decentralized – every node in a pipeline can be easily distributed across multiple servers with a minimal change to the pipeline definition and no change to the node's updater. The approach has been implemented in the RDF Pipeline Framework[67], an open source project. Nevertheless, as stated by the authors, the RDF Pipeline Framework is not yet ready for general production release; on the contrary, maturity of UnifiedViews has been proved in numerous projects. Furthermore, the RDF Pipeline Framework provides no user interface for managing, monitoring, or debugging pipelines, managing individual plugin nodes, or getting notifications about pipelines' executions. On the other hand, RDF Pipeline Framework describes pipelines using RDF data model; in UnifiedViews, we also plan in the future to support descriptions of pipelines using RDF data model.

Rautenberg et al. [11] present LODFlow, a Linked Data workflow management system, which provides an environment for planning, executing, reusing, and documenting Linked Data workflows. Nevertheless, the authors focus mainly on the description of a comprehensive ontological model, the Linked Data Workflow Project Ontology, for describing the workflows and a workflow execution engine, but the actual implementation of the workflow system is an ongoing and mainly future work[68].

Open Refine[13] is a tool for cleansing, transforming, and enriching tabular data. It has also RDF extension which provides a service to disambiguate cell values to Linked Data entities, e.g., from DBpedia knowledge base. Nevertheless, when compared with UnifiedViews, the purpose of such tool is different - it is used for manual refinement of the data, whereas UnifiedViews is used for preparing tasks, which may be executed repeatedly without the interaction of the user.

LinkedPipes ETL (LP-ETL) [7] is an RDF based ETL framework being developed based on the experience gathered while publishing Linked Data using UnifiedViews with a similar set of features. Compared to UnifiedViews, LP-ETL does not provide features such as using relational data units in the pipelines, multilingual user interfaces and granular user and permission management. LP-ETL refocuses on Linked Data and the definition of the pipelines themselves. From the technical point of view, LP-ETL uses RDF as a native format for storage of pipelines and configuration, it facilitates sharing of pipelines and their fragments directly using their URLs and URL dereferencing and it has a more granular, RESTful API providing access to all functionality. From the user point of view, the process of pipeline creation is more intuitive as algorithms for suggesting next possible DPUs based on various factors are employed, and sample pipeline fragments are attached and ready to be reused directly in the documentation of individual DPUs. UnifiedViews pipelines can be imported to LP-ETL given that the unsupported features are not used.

[63] http://grafter.org/
[64] http://clojure.org/
[65] https://datagraft.net/
[66] http://opendatanode.org/
[67] https://github.com/rdf-pipeline
[68] https://github.com/AKSW/LODFlow/

6. Conclusions

We presented UnifiedViews, an open-source ETL framework for processing RDF data, which addresses the problem of efficiently creating, debugging, and maintaining Linked Data processing tasks.

UnifiedViews combines several aspects that are crucial to be successful within these projects:

- Support for complex data processing tasks which may contain forking and merging of data flows; as a result, complex RDF data processing flows combining data from different sources may be easily prepared in UnifiedViews.
- A robust data processing engine; this was proofed, e.g., by Opendata.cz project involving preparation and regular execution of tens of pipelines processing milions of triples (see Table 1).
- Flexibility to choose from a large number of existing DPUs – UnifiedViews provides 35+ core DPUs and tens of DPUs provided by UnifiedViews team[69].
- A simple way to create custom DPUs by providing an extensive set of tutorials[70] for DPU developers.
- RDF data debugging. Data wrangles may debug (RDF) data flows between DPUs as they are preparing pipelines, which decreases the time needed to prepare pipelines.
- An intuitive user interface, which was confirmed by data wranglers from most of the projects in Section 3, e.g., the project with *Council Open Data Initiative*.

Exemplary use cases introduced in Section 3 also allowed us to reveal certain shortcoming of UnifiedViews, which were summarized in Section 4.

UnifiedViews is pushed forward by a unique collaboration of a diverse group of partners – research institutes and SME's across Europe[71]. As a result, UnifiedViews has ascended from a research prototype to a mature toolkit with applications in research and innovation projects as well as in an commercial context.

[69] http://unifiedviews.eu/#get-team

[70] https://grips.semantic-web.at/pages/viewpage.action?pageId=50929588

[71] http://unifiedviews.eu/#get-team

References

[1] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1 – 22, 2009.

[2] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. Dbpedia - a crystallization point for the web of data. *Web Semant.*, 7(3):154–165, Sept. 2009.

[3] D. Booth. The RDF pipeline framework: Automating distributed, dependency-driven data pipelines. In *Data Integration in the Life Sciences - 9th International Conference, DILS 2013, Montreal, QC, Canada, July 11-12, 2013. Proceedings*, pages 54–68, 2013.

[4] S. H. Garlik, A. Seaborne, and E. Prud'hommeaux. SPARQL 1.1 Query Language. W3C Recommendation, 2013. http://www.w3.org/TR/2013/REC-sparql11-query-20130321/, Retrieved 20/03/2014.

[5] P. Gearon, A. Passant, and A. Polleres. SPARQL 1.1 Update. W3C Recommendation, 2013. http://www.w3.org/TR/2013/REC-sparql11-update-20130321/, Retrieved 20/03/2014.

[6] J. Klímek, J. Helmich, and M. Nečaský. Use Cases for Linked Data Visualization Model. In *Proceedings of the Workshop on Linked Data on the Web, LDOW 2015, co-located with the 24th International World Wide Web Conference (WWW 2015), Florence, Italy, May 19th, 2015.*, 2015.

[7] J. Klímek, P. Škoda, and M. Nečaský. LinkedPipes ETL: Evolved Linked Data Preparation. In *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 95–100, 2016.

[8] T. Knap, J. Michelfeit, J. Daniel, P. Jerman, D. Rychnovský, T. Soukup, and M. Nečaský. ODCleanStore: A Framework for Managing and Providing Integrated Linked Data on the Web. In X. S. Wang, I. F. Cruz, A. Delis, and G. Huang, editors, *WISE*, volume 7651 of *Lecture Notes in Computer Science*, pages 815–816. Springer, 2012.

[9] T. Knap, J. Michelfeit, and M. Necaský. Linked Open Data Aggregation: Conflict Resolution and Aggregate Quality. In *COMPSAC Workshops*, pages 106–111, Izmir, Turkey, 2012. IEEE Computer Society.

[10] D. L. Phuoc, A. Polleres, M. Hauswirth, G. Tummarello, and C. Morbidoni. Rapid prototyping of semantic mash-ups through semantic web pipes. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, *WWW*, pages 581–590. ACM, 2009.

[11] S. Rautenberg, I. Ermilov, E. Marx, S. Auer, and A.-C. Ngomo Ngonga. Lodflow – a workflow management system for linked data processing. In *SEMANTiCS 2015*, 2015.

[12] A. Schultz, A. Matteini, R. Isele, C. Bizer, and C. Becker. LDIF : Linked Data Integration Framework. In *Proceedings of the Second International Workshop on Consuming Linked Data (COLD)*, Bonn, Germany, 2011. CEUR-WS.org.

[13] R. Verborgh and M. De Wilde. *Using OpenRefine*. Packt Publishing, Sept. 2013.

[14] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov. Silk - A Link Discovery Framework for the Web of Data. In *Proceedings of the WWW2009 Workshop on Linked Data on the Web (LDOW)*, Madrid, Spain, 2009. CEUR-WS.org.