UnifiedViews: An ETL Tool for RDF Data Management

Tomáš Knap[1], Peter Hanečák[3], Jakub Klímek[1], Christian Mader[2], Martin Nečaský[1],
Bert Van Nuffelen[4], and Petr Škoda[1]

[1] Charles University in Prague, Faculty of Mathematics and Physics
Malostranské nám. 25, 118 00 Praha 1, Czech Republic
{surname}@ksi.mff.cuni.cz

[2] Semantic Web Company
Mariahilfer Straße 70 / 8
A - 1070 Vienna, Austria
c.mader@semantic-web.at

[3] EEA s.r.o
Hattalova 12B, 831 03 Bratislava, Slovak Republic
peter.hanecak@eea.sk

[4] TenForce bvba
Havenkant 38, 3000 Leuven, Belgium
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 differs from other ETL frameworks by natively supporting management 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 directions in which UnifiedViews is exploited. Based on our practical experience with the tool, we found that UnifiedViews contributes to simplifying the task for data providers to establish and maintain Linked Data publication processes.

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[5] 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, and (5) solving data conflicts to prepare the integrated data mart.

[5] https://en.wikipedia.org/wiki/Data_mart

There are numerous tools used by the Linked Data community[6], which may support various phases of the data mart preparation; e.g., the wrangler may use *any23*[7] to extract non-RDF data and convert such data to RDF data format, *OpenLink Virtuoso*[8] database for storing RDF data and executing SPARQL (Update) queries [3, 4], *Silk* [10] to interlink RDF data based on the declarative rules, or *Cr-batch* [6] to solve RDF data conflicts[9]. Nevertheless, using such tools directly, the data wrangler has to (1) configure every such tool differently, (2) write a custom script downloading and unpacking data from various data sources, (3) prepare a script executing a set of SPARQL Update queries curating the data, (4) implement custom transformers which, e.g., enrich processed data with the data from the DBpedia knowledge base[10], (5) 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*[11], 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 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, the data flow etc; in contrary, if there were graphical visualizations of the prepared tasks, which shows tools being used and the data flow between these tools, the data processing task would be documented and maintenance 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[12], an open-source Extract-Transform-Load (ETL) framework. It is an integrated solution that provides standard maintenance interfaces and lets data

[6] http://semanticweb.org/wiki/Tools

[7] https://any23.apache.org/

[8] http://virtuoso.openlinksw.com/

[9] https://github.com/mifeet/cr-batch

[10] http://wiki.dbpedia.org/

[11] http://linux.die.net/man/8/cron

[12] http://unifiedviews.eu

wranglers choose from various pre-defined and customizable „bulding blocks" to set up the individual data processing steps.

This paper is organized as follows: in Section 2, we present (1) basic concepts of UnifiedViews, (2) how data wrangler may interact with UnifiedViews, and (3) architecture of UnifiedViews. We provide an outline of related work in this problem domain in Section 3. To demonstrate the maturity of UnifiedViews, we introduce in Section 4 a number of exemplary projects in which the framework is successfully used. We outline current research projects/directions in which UnifiedViews is exploited in Section 5 and draw our conclusions in Section 6.

2 UnifiedViews Framework

In this section we introduce the basic concepts of UnifiedViews, describe how data wranglers may interact with UnifiedViews, and elabaorate on the architecture of the framework. We also discuss the availability of UnifiedViews.

2.1 Basic Concepts of UnifiedViews

A data processing task, such as the one in our introductory example above (preparation of a data mart by the wrangler), is represented in UnifiedViews as a *data processing pipeline* (or simply *pipeline*). 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 the DPU may be configured by a pipeline designer (e.g., the data wrangler mentioned above); administrators of the particular UnifiedViews installations may set up the default configurations of such DPUs and also prepare various alternative configurations of the DPUs, which may be directly reused 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 DPU may declare $0 - N$ input data units and $0 - N$ 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 manage:

- *RDF* data units which hold RDF graphs,
- *Files* data units for accessing local files, and
- *Relational* data units for holding tables from relational databases.

Every data unit can hold $0 - N$ entries of the particular datatype it supports.

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:

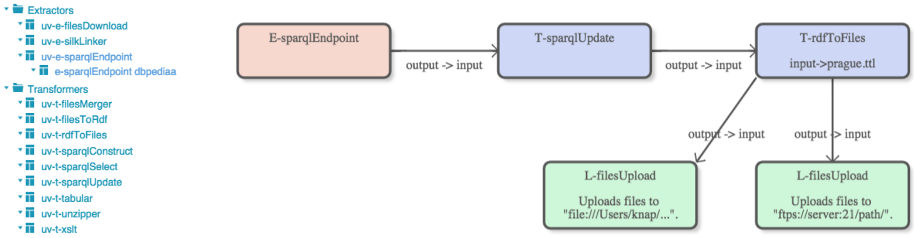

Fig. 1. UnifiedViews Framework – Definition of a Data Processing Task

- *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[13] 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.

2.2 Interacting with UnifiedViews

UnifiedViews provides a graphical user interface, which allows users (e.g., data wranglers) to define, maintain, execute, monitor, debug, schedule, and share DPUs and pipelines. Figure 1 depicts a screenshot of this interface. It shows a data processing pipeline, consisting of five DPUs (colored boxes) and four data flows (arrows connecting the boxes) between these DPUs. DPUs may be added and moved by drag&drop on a canvas and 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.

During preparation of the pipeline, UnifiedViews provides users with debugging capabilities. A user may execute a selected fragment of the pipeline at any time and

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

browse or query (using the SPARQL query language) the entries in the input- and output data units that are consumed or produced by each DPU.

When users 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. Users 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 execution states of selected pipelines in the last 24 hours.

UnifiedViews currently provides more than 30 *core DPUs*[14]. These are available for users 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 Update queries, or executing XSL transformations, and
- loading the transformed and curated data to external systems.

Apart from that, the UnifiedViews team[15] also provides additional DPUs. If a specific functionality is required that is not covered by a DPU available among the core DPUs or those provided by the team, users may easily create and deploy their own custom DPUs; for this purpose, extensive documentation such as tutorials is provided online[16].

2.3 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[17] framework.
- *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, maintain, 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.

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

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

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

[17] http://vaadin.com

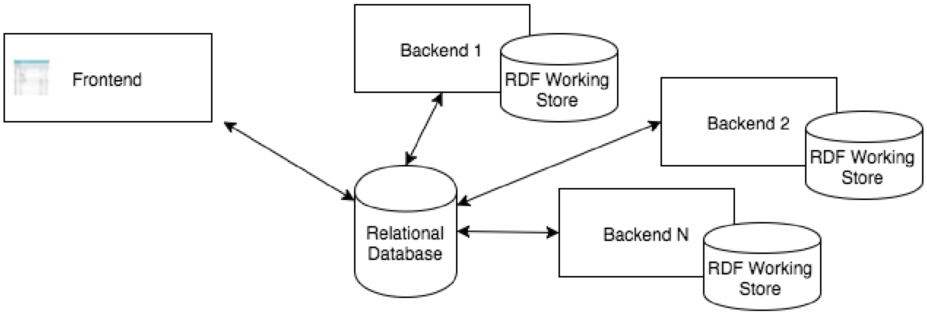

Fig. 2. UnifiedViews Framework – Frontend, Backend

Frontend and backend communicate via a relational database, which stores all configuration information such as pipeline setups, DPU configurations, execution states, or scheduled events (see Figure 2). To support scalability, multiple backend instances can run on different machines, effectively executing pipelines in parallel (see Figure 2).

Every backend uses its own *RDF Working Store* for storing temporary data which is produced by the pipeline during its execution (see Figure 2). As the RDF Working Store, we currently support Sesame[18] 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, can be used as an RDF Working Store.

Every DPU is an OSGi[19] 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 user wants to interact (view, edit, save, delete, etc.) with a certain entity (pipeline, DPU, scheduled event, etc.), UnifiedViews checks whether the user is authorized to do so. Spring Security[20] is used to ensure authentication and authorization of users.

2.4 Availability of UnifiedViews

The source code of UnifiedViews is published under an open-source license which is a combination of GPLv3[21] and LGPLv3[22] licenses. It is hosted at GitHub[23] and divided into the following repositories:

[18] `http://rdf4j.org/`
[19] `https://www.osgi.org/`
[20] `http://projects.spring.io/spring-security/`
[21] `http://www.gnu.org/licenses/gpl.txt`
[22] `http://www.gnu.org/licenses/lgpl.txt`
[23] `https://github.com/UnifiedViews`

- *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[24]. For a quick start, UnifiedViews can be also easily explored using the Docker[25] and Vagrant[26] instances.

3 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[27] – for example Clover ETL (community edition)[28], KETL[29], or EplSite ETL[30]. 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 UnfiedViews, we focus on the area of RDF ETL frameworks in the remainder of this section.

ODCleanStore[31] [5] is a Java based Linked Data Management framework developed at Charles University in Prague, Department of Software Engineering. Linked Data Manager (LDM)[32] 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 in Austria[33].

[24] http://unifiedviews.eu

[25] https://github.com/tenforce/docker-unified-views

[26] https://github.com/tenforce/vagrant-unifiedviews2.x-ubuntu15.04

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

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

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

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

[31] https://github.com/mff-uk/ODCS/

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

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

As part of LOD2 project[34], teams responsible for ODCleanStore and LDM decided to develop a common tool, UnifiedViews, described in this paper, which was designed from the beginning to supersede ODCleanStore and LDM tools; it is inspired by both these tools and it improves in particular (1) the robustness of the pipeline execution engine, (2) usability of the graphical user interface, and (3) simplicity of new DPUs' creation.

DERI Pipes [7] 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) [9] 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[35]. 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[36].

Grafter[37] 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[38] language is used to prepare pipeline definitions. DataGraft[39] is a graphical user interface for definining 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 Closure functions. When comparing UnifiedViews and Grafter/DataGraft, the latter one provides supports only for transforming tabular data (CSV/Excel) to Linked Data; on the other hand, UnifiedViews provides 30+ 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

[34] `http://stack.lod2.eu/blog/`

[35] `http://ldif.wbsg.de/`

[36] `http://hadoop.apache.org/`

[37] `http://grafter.org/`

[38] `http://clojure.org/`

[39] `https://datagraft.net/`

available in UnifiedViews, nevertheless it is covered by Open Data Node[40]. Grafter is also not that matured as UnifiedViews, it is under active development and authors plan breaking changes in the next future.

Booth [2] 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[41], 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. [8] 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[42].

4 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 (1) the motivation and goals, (2) approach we took and achievements we reached, and (3) challenges we faced and lessons learned.

4.1 The Czech Trade Inspection Authority

The Czech Trade Inspection Authority in Czech Republic (CTIA)[43] examines and monitors fairness of businesses which supply or sell goods, provides services, or operates marketplaces in Czech Republic.

Motivation and Goals. Before CTIA publishes their core datasets – data about inspections, bans, and sanctions – as open data, lots of subjects (citizens, companies) requested

[40] `http://opendatanode.org/`

[41] `https://github.com/rdf-pipeline`

[42] `https://github.com/AKSW/LODFlow/`

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

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[44] and CTIA has to always 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.

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[45]. The datasets are available at their official web site[46]. 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[47][48], which confirms there is a public demand in CTIA data and it also directly shows the benefits of having data published as Linked Data.

Challenges and Lessons Learned. We find out that there are governmental institutions willing to publish their data as (Linked) open data and UnifiedViews is able to effectively help them realizing 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; however, CTIA reported that it was easy and intuitive for them to further maintain the data processing pipelines in UnifiedViews.

CTIA data workers had two issues when creating pipelines in UnifiedViews themselves, without any consultation – (1) they are 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.

4.2 The Czech Social Security Administration

The Czech Social Security Administration (CSSA)[49] collects and enforces payable social security premiums, which includes pension insurance, sickness insurance, and a

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

[45] http://opendata.cz/en

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

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

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

[49] http://www.cssz.cz/en

contribution to the state employment policy. The CSSA takes decisions on most of the pension benefits and arranges to pay them.

Motivation and Goals. CSSA decided to publish their internal data sources – statistical data about pensions (mostly Excel files) – as open data. The primary motivation for that was similar as the one introduced in Section 4.1 in case of CTIA project – to lower the costs for an ad hoc creation of data reports by publishing their data upfront in a machine readable form. They decided to publish the data as Linked Data (in RDF data format) and also as CSV data. Apart from that, they also wanted to manage their published open data in the catalog.

Approach and Achievements. UnifiedViews was used to transform Excel files to both RDF and CSV formats; in case of RDF data format, Data Cube vocabulary[50] was used to publish statistical data about pensions. Further, UnifiedViews was exploited to publish the transformed data to the official CSSA open data catalog[51]. UnifiedViews also ensured that the published data is accompanied with DCAT-AP[52] compatible metadata.

Challenges and Lessons Learned. After the initial consultation provided by Charles University in Prague, Department of Software Engineering, CSSA data workers were able to create their own pipelines in UnifiedViews. Nowadays, these UnifiedViews pipelines regularly transforming and publishing pensions data are operated by the CSSA data workers, who really like the simplicity of the maintenance. The challenge we faced was the repetition of the same configuration parameters throughout the single UnifiedViews pipeline for the given dataset; to solve this issue, we plan to extend UnifiedViews (1) to support parametrization of pipelines and also (2) having the possibility to pack fragments of pipelines as DPUs and being able to place such pipeline fragments inside other pipelines.

4.3 Council Open Data Initiative

The Council of the Europian Union (EU Council) is, together with the European Parliament, the legislative body of the EU. Its voting records are public whenever it adopts a legislative act, so that citizens can see how each country has voted.

Motivation and Goals. Until now, 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.

[50] http://www.w3.org/TR/vocab-data-cube/

[51] https://data.cssz.cz (in Czech)

[52] https://joinup.ec.europa.eu/asset/dcat_application_profile/description

The goal of the project (which is currently ongoing) 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.

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 Virtuoso RDF store. To achieve this, Semantic Web Company, realizing this project for EU Council, developed a multi-step pipeline which is scheduled and bi-hourly executed. It consists of an extraction step, multiple transformations and the final loading (storage) stage. The pipeline creates 311,000 RDF Triples, using 23 classes, 45 different predicates and about 18,000 different subjects.

As a first example of date re-use, three data visualizations have been created: a map visualization[53], a visualization of votes over time[54] and votes on a punch-card[55]. To ensure currency, these visualizations are always created from the data directly taken from the Council Open Data Initiative's API.

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 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. Although Semantic Web Company provided the main DPUs and pipelines needed for achieving the project's goals, Semantic Web Company was able to instruct the council members to maintain (i.e., control and observe) the pipelines via email. Prior to this project, the council had only minor experience with Linked Data. We can therefore conclude that UnifiedViews, as a user-friendly graphical tool has tremendous value for encouraging institutions to publish their data in an open format on the Web, contributing to increased availability of high quality Linked Data.

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

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

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

4.4 Open Data Support: First pan-European Open Data Portal

The European Commission Directorate General for Communications Networks, Content & Technology (DG Connect)[56] 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[57] and the European Open Data Forum[58].

Motivation and Goals. DG Connect launched a 3-year project Open Data Support[59] 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.

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[60]. DCAT-AP harmonizes and adopts the W3C DCAT specification[61] 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 data. The Open Data Support harvesting pipeline is the first commercial application of UnifiedViews.

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 core 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

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

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

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

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

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

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

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.

We find out that support for Virtuoso as RDF Working Store in UnifiedViews would be beneficial to this project in order to achieve better throughput performance.

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.

4.5 Westtoer Datahub

Westtoer[62] 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.

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[63]. 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.

Approach and Achievements. For the Westtoer datahub UnifiedViews is deployed as a dockized solution exporting the data to a Virtuoso RDF store. Via the DataTank[64] end-users can access the data.

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 faster than anticipated.

[62] `http://www.westtoer.be`

[63] `http://datahub.westtoer.be`

[64] `http://thedatatank.com/`

Challenges and Lessons Learned. The creation of the UnifiedViews pipelines was a more labor intense work, mostly because of the unfamiliarity with the source and target vocabularies.

The pipelines, which are based entirely on SPARQL (Update) queries, are not always that performant when the update size increases. Solutions such as splitting up the input files are needed to reduce the update search size.

We observed also that pipelines based on a JSON input execute faster that those based on XML input. From our first assessments this is due the XML parsing takes more time and also because the XML pipeline contains more blank nodes which have to be dealt with.

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.

4.6 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.

Motivation and Goals. SEA wanted to explore the potential to increase re-use 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[65]. 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.

Approach and Achievements. UnifiedViews was successfully deployed (as one of components of Open Data Node (ODN) publication platform[66]) on the remote cloud infrastructure of SEA. During pilot collaboration with COMSODE project[67], 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[68] and datasets from the European Environmental Agency: Biogeographical regions 2011, Natura 2000 and EUNIS[69]. The results of the transformations were published using other components of

[65] http://inspire.ec.europa.eu/index.cfm/pageid/2/list/datamodels
[66] http://opendatanode.org/
[67] http://www.comsode.eu/
[68] http://www.geonames.org/
[69] http://www.eea.europa.eu/data-and-maps

ODN: UnifiedViews loaders, CKAN catalog and LDVMi visualization framework[70]. The catalog with published data is available online[71].

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.

4.7 OpenData.cz Initiative

OpenData.cz initiative[72] is the initiative of a group of people mainly from Charles University in Prague and University of Economics, Prague.

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.

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[73].

Challenges and Lessons Learned. Using UnifiedViews to maintain data processing tasks of OpenData.cz initiative was really effective and it simplified our data processing tasks and at the same time kept our data processing tasks documented.

OpenData.cz initiative realized that certain fragments of pipelines tend to repeat for many pipelines, 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. This finding confirms the lesson learned from the CSSA project described in Section 4.2.

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.

[70] `http://opendatanode.org/product/ldvmi-visualisation/`

[71] `http://data.sazp.sk/`

[72] `http://opendata.cz`

[73] `http://linked.opendata.cz/en`

5 UnifiedViews in Research Projects

Currently, it is planned to make use of UnifiedViews in several research projects.

The goal of the EU-funded ALIGNED project[74] 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[75] 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.

Within the EU-funded YourDataStories[76] (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 4, 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.

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 for serving transformation and/or publication use cases.

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

- the ability to create complex RDF data processing flows combining data from different sources,
- the flexibility to choose from a large number of existing DPUs,
- the simplicity of custom DPUs' creation,
- the attention towards RDF data debugging and error handling,

[74] `http://aligned-project.eu/`

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

[76] `http://yourdatastories.eu/`

- the right mixture of predefined, out of the box functionality, with flexible adaptable transformation scripting, and
- the intuitive user interface.

UnifiedViews has ascended from a research prototype to a mature toolkit being applied both in (1) research and innovation projects and (2) commercial context. We support this claim by having introduced a set of exemplary use cases in Section 4 where UnifiedViews was successfully deployed.

Exemplary use cases introduced in Section 4 also allowed us to reveal certain shortcoming of UnifiedViews, which we will try to address in further research projects; currently starting research projects are listed in Section 5.

UnifiedViews is currently pushed forward by a unique collaboration of a diverse group of partners – research institutes and SME's across Europe[77].

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

Bibliography

[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] 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.

[3] 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.

[4] P. Gearon, A. Passant, and A. Polleres. SPARQL 1.1 Update. Technical report, W3C, 2013. Published online on March 21st, 2013 at `http://www.w3.org/TR/2013/REC-sparql11-update-20130321/`, Retrieved 20/03/2014.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.