PlanetData

Network of Excellence

FP7 – 257641

D5.4 PlanetData data management tools catalogue and access portal

Coordinator: Oscar Corcho (UPM)

With contributions from: EPFL, INRIA, JSI, FUB, SOTON

1st Quality Reviewer: Serge Tymaniuk
2nd Quality Reviewer: Max Schmachtenberg

Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	M48
Actual delivery date:	M48
Version:	1.0
Total number of pages:	25
Keywords:	tool catalogue, ADMS

Abstract

One of the objectives of PlanetData will be the generation of a catalogue of tools and services that can be applied in the context of Big/Open/Linked Data. Previous deliverables focused mainly on Linked and Sensor Data tools and APIs, for which information was collected from PlanetData core and associated members. This deliverable extends the previous catalogue with revised descriptions of these tools, platforms and frameworks, and with tools that were not covered in the previous deliverable. Contributions to this catalogue have been done by PlanetData core and associated members. This tool catalogue is organized and published as part of the PlanetData portal in WP7. More precisely, we describe metadata of each tool as well as information about the tools according to this metadata. As in the previous deliverable, D5.3, this metadata is largely based on the ADMS vocabulary (`http://www.w3.org/ns/adms`), proposed in the context of the EU JoinUp initative.

Executive Summary

In this deliverable, we have revised the descriptions of the tools that had been already described in D5.3, with nine main groups of metadata items (according to ADMS vocabularies): general information, language, documentation, contact, status, item, distribution, license, and publisher. These tools were collected from PlanetData core and associated members through a Google Docs form, and have been revised afterwards to reflect their current status. The current list of tools from PlanetData core and associated members include:

- GSN (EPFL)
- MonetDB (CWI)
- LDIF, D2RQ (University of Mannheim)
- linked-data-fu (KIT)
- morph-rdb, morph-gft, morph-ldp, morph-streams, geometry2rdf, OOPS! (UPM)
- IJS Newsfeed, Videk, LODMiner (JSI)
- ckanext-silk, ckanext-sparql, ckanext-metadata, ckanext-extractor (Universidad de Deusto)
- Rhizomer (Universitat de Lleida)
- HDT (University of Valladolid)
- Yet Another SPARQL GUI (VU University)
- Datalift (INRIA)

It is important to highlight that this report only serves as a summary of the classified tools, and that this report is published in a systematic and structured manner in the dissemination platform of WP7.

Document Information

IST Project Number	FP7 – 257641	Acronym	PlanetData
Full Title	PlanetData		
Project URL	http://www.planet-data.eu/		
Document URL	http://planet-data.eu/sites/default/files/PD D5.4 Tools Catalogue.pdf		
EU Project Officer	Leonhard Maqua		

Deliverable	Number	D5.4	Title	PlanetData data management tools catalogue and access portal
Work Package	Number	WP5	Title	PlanetData Lab

Date of Delivery	Contractual	M48	Actual	M48
Status	version 1.0		final ☒	
Nature	Report (R) ☒ Prototype (P) ☐ Demonstrator (D) ☐ Other (O) ☐			
Dissemination Level	Public (PU) ☒ Restricted to group (RE) ☐ Restricted to programme (PP) ☐ Consortium (CO) ☐			

Authors (Partner)	Oscar Corcho (UPM)			
Responsible Author	Name	Oscar Corcho	E-mail	ocorcho@fi.upm.es
	Partner	UPM	Phone	+34913366605

Abstract (for dissemination)	One of the objectives of PlanetData will be the generation of a catalogue of tools and services that can be applied in the context of Big/Open/Linked Data. Previous deliverables focused mainly on Linked and Sensor Data tools and APIs, for which information was collected from PlanetData core and associated members. This deliverable extends the previous catalogue with revised descriptions of these tools, platforms and frameworks, and with tools that were not covered in the previous deliverable. Contributions to this catalogue have been done by PlanetData core and associated members. This tool catalogue is organized and published as part of the PlanetData portal in WP7. More precisely, we describe metadata of each tool as well as information about the tools according to this metadata. As in the previous deliverable, D5.3, this metadata is largely based on the ADMS vocabulary (`http://www.w3.org/ns/adms`), proposed in the context of the EU JoinUp initative.

Keywords	tool catalogue, ADMS

Version Log			
Issue Date	**Rev. No.**	**Author**	**Change**
05/04/2013	0.1	Nguyen Quoc Viet Hung	Final version of deliverable D5.3
01/09/2014	0.2	Oscar Corcho	Added, revised and corrected metadata of previous tool descriptions plus new tools included in this deliverable
09/09/2014	0.5	Oscar Corcho	Added information of all tools
30/09/2014	1.0	Oscar Corcho	New version according to reviews received

Table of Contents

Abbreviation

ADMS	Asset Description Metadata Schema
DCMI	Dublin Core Metadata Initiative
GML	Geography Markup Language
WKT	Well-known text

1 Introduction

As the number of data are likely to grow at large, data management tools have become essential for the management of systems and infrastructure. One of the objectives of the PlanetData Lab (WP5) is to provide a collection of structured and systematic descriptions of existing data management tools, so that they can serve as a reference for researchers and practitioners working in this area. Such descriptions are based on a set of metadata fields that allow better indexing, searching, and browsing through the information about the collected tool descriptions. All the tool descriptions are categorized according to such proposed metadata and their information is published on the dissemination platform developed in WP7.

Our metadata set extends the ADMS vocabulary[1], which has been proposed in the context of the EU JoinUp initiative. This metadata set was already described in deliverable D5.3. However, there have been several changes since that version, what has forced us to regenerate all data according to it.

In this document, each tool, platform or framework is categorized according to the proposed metadata, with the aim of helping PlanetData partners and any other readers in the use of the listed tools by explicit links to their documentation and use cases.

The deliverable is organized as follows. Chapter 2 summarizes the metadata used for tool catalogue according to ADMS vocabularies. This section is largely based on the work that was already presented in D5.3, but now takes into account all changes that have been done to ADMS, and which have been implemented in the RDF data that we have generated. Chapter 3 discusses the details of the updated tool catalogue, together with up-to-date information about the collected tools. Chapter 4 is dedicated to the summarization of the tools and concludes the deliverable.

[1] `http://www.w3.org/ns/adms`

2 Metadata

For the purpose of the creation of our tool catalogue, we have decided to use the ADMS vocabulary `http://www.w3.org/ns/adms` as our core metadata set. This metadata helps in organizing the collected tools more logically and search across them more efficiently. We have opted for ADMS since it is a standard vocabulary proposed by the European Commission Joinup initiative [1].

In this section we provide a brief description of the part of ADMS that we have decided to use for our publication. We use the following prefixes in our descriptions:

1. adms: <http://www.w3.org/ns/adms#>
2. dcat: <http://www.w3.org/ns/dcat#>
3. dcterms: <http://purl.org/dc/terms/>
4. skos: <http://www.w3.org/2004/02/skos/core#>
5. vcard: <http://www.w3.org/2006/vcard/ns#>

We are using the following classes from ADMS: adms:AssetRepository (used to describe the whole repository), adms:Asset (used to describe tools and platforms), adms:AssetDistribution (to refer to specific distributions of the tools and platforms), dcterms:LicenseDocument (to refer to licenses of use of the tools and platforms), and vcard:VCard (for contact information about the owners or distributors of the tool or platform). We will now provide a description of these main concepts and the properties that we are using for each of them, while Figure 2.1 provides a graphical overview of these and the other classes that are not used in our repository.

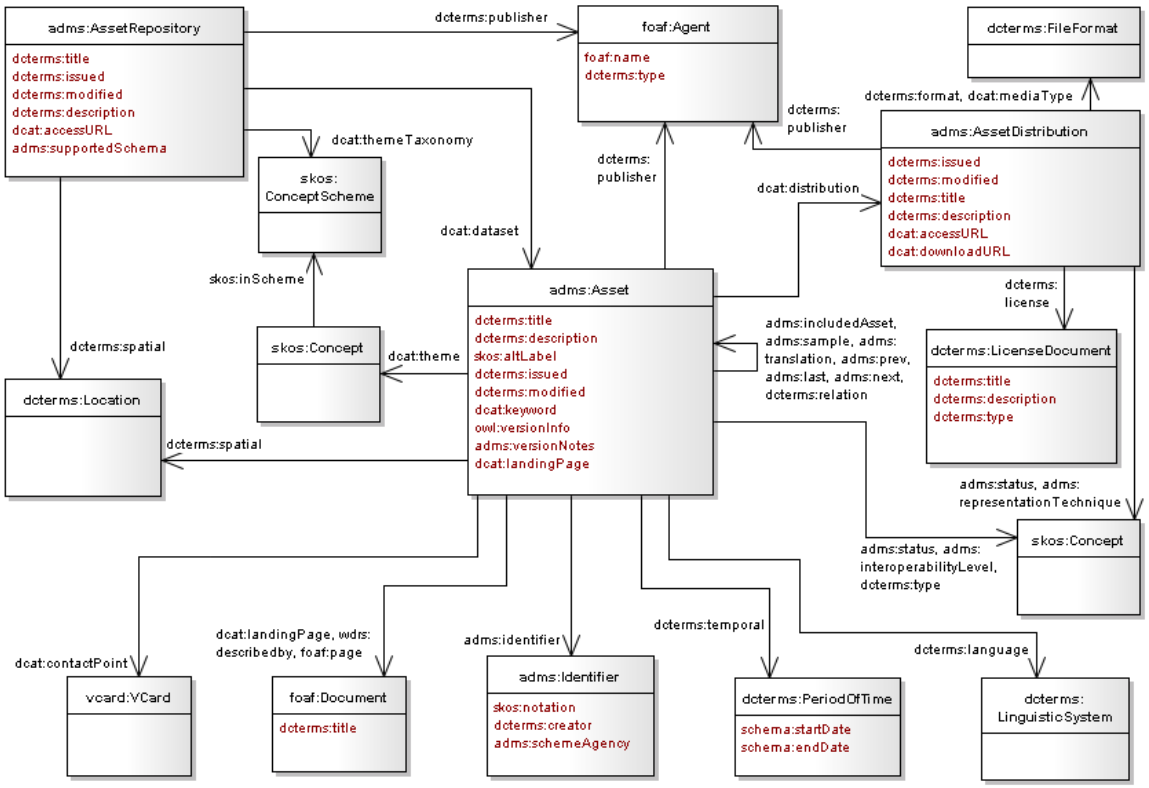

Figure 2.1: Main classes and properties of the ADMS vocabulary (source: ADMS documentation)

[1] `https://joinup.ec.europa.eu/`

2.1 AssetRepository

The concept AssetRepository is used to provide general information about the whole repository. The properties that are applied to asset repositories provide a broad overview and a general description about them. The properties that we have selected for our repository are defined either in ADMS, Dublin Core or DCAT, and include:

1. Title (dcterms:title)
2. Description (dcterms:description)
3. Date of creation (dcterms:issued)
4. Date of last modification (dcterms:modified)
5. Access URL (dcat:accessURL)
6. Publisher (dcterms:publisher)
7. DataSet (dcat:dataset). This property allows connecting the AssetRepository with the different assets that belong to it.

2.2 Asset

The concept Asset is used to provide general information about every tool and platform. The properties that are applied to assets provide a broad overview and a general description about them. The properties that we have selected for our repository are defined either in ADMS, Dublin Core or DCAT, and include:

1. Title (dcterms:title)
2. Description (dcterms:description)
3. Alternative name (skos:altLabel)
4. Date of creation (dcterms:created)
5. Date of last modification (dcterms:modified)
6. Publisher (dcterms:publisher)
7. Keywords (dcat:keyword)
8. Version (adms:last)
9. Version notes (adms:versionNotes)
10. Landing page (dcat:landingPage)
11. Status (adms:status). This property can take as a value any of the following SKOS concepts: completed, deprecated, under development or withdrawn.
12. Distribution (dcat:distribution). This property allows connecting an asset with its different distributions, which are instances of the class AssetDistribution.
13. Contact point (dcat:contactPoint). This property allows connecting an Asset with the corresponding contact point, described as a VCard.

2.3 AssetDistribution

This concept is used to represent a particular release of a tool or platform, which can used as a fully-functional software for a particular purpose. We have decided to represent only the latest distribution of each platform and tool, at the time of releasing this catalogue. A distribution is typically a downloadable computer file that implements the specific requirements of a tool. Each distribution is associated with only one tool. A distribution in our catalogue is described with the following properties:

1. Title (dcterms:title)
2. Description (dcterms:description)
3. Date of creation (dcterms:issued)
4. Date of last modification (dcterms:modified)
5. Representation technique (adms:representationTechnique): the range of this attribute falls into one of following values: Archimate/ BPMN/ CommonLogic/ DTD/ Datalog/ Diagram/ Genericcode/ Human-Language/ IDEF/ KIF/ OWL/ Prolog/ RDFSchema/ RIF/ RelaxNG/ RuleML/ SBVR/ SKOS/ SPARQL/ SPIN/ SWRL/ Schematron/ TopicMaps/ UML/ WSDL/ WSMO/ XMLSchema/ other.
6. Format (dcterms:format): indicates the technical computer format in which the distribution is available. Technically, ADMS proposes the use of the dcterms:FileFormat class to fully represent this attribute.
7. Access URL (dcat:accessURL)
8. Download URL (dcat:downloadURL)
9. Status (adms:status). The values of this property are the same as the ones described for the class Asset.

2.4 VCard

This class provides the contact information of the person responsible for a developed tool, such as name, e-mail, address, etc. A contact information card in our catalogue includes, but not limited to, the following important attributes:

1. Name (vcard:name)
2. E-mail (vcard:email)
3. Full address (vcard:address)
4. Telephone (vcard:phone)
5. Web page (vcard:web)

2.5 LicenseDocument

The LicenseDocument class is used to describe the license that is applicable to the tool. There are various types of licenses, such as GPL, LGPL, Apache, and MIT. A license has the following properties:

1. Name (dcterms:title)
2. Description (dcterms:description)
3. Type (dcterms:type)

3 Tool Catalogue

In this section, we provide a summary of the tools that we have collected in our catalogue. We start by providing the RDF code that corresponds to out AssetRepository, and which contains links to every tool description (Section 3.3). Then we offer the current list of descriptions of these tools, in Section 3.2. Then, Section 3.3 describes how we have structured the tool catalogue according to three important dimensions: by input data, by functionality, and by representation technique (which allow providing different views over the catalogue in the PlanetData website). For each dimension, we consider the most relevant categories to reflect the properties of each tool or platform.

3.1 The PlanetData Tool Catalogue described as an Asset Repository

This section contains the RDF code that we have created for the description of the tool catalogue as an adms:AssetRepository instance. It contains the following information:

```
:PlanetDataCatalogue a adms:AssetRepository ;
  dcterms:created "2013-05-31"^^xsd:date ;
  dcterms:modified "2014-09-30"^^xsd:date ;
  dcterms:description "A catalog of tools coming from PlanetData core and
     associated members"@en ;
  dcterms:publisher <http://www.planet-data.eu/> ;
  dcat:accessURL <http://planet-data.eu/planetdata-tool-catalogue> ;
  dcterms:title "PlanetData Data Management Tool Catalogue"@en ;
  dcat:dataset :GSN ;
  dcat:dataset :HDT ;
  ...
  dcat:dataset :Rhizomer.
```

3.2 Individual Descriptions of the PlanetData Tool Catalogue

In this section, we provide a summary of the metadata that we have collected for each tool (the complete set of metadata is available on the tool catalogue that has been made available on the PlanetData website). For the sake of simplicity, the listings contain the following key aspects:

- Description (description of the tool)
- Date of creation (the date of creation of the tool source code)
- Date of last modification (the date of last changes to the tool source code)
- Keywords
- Status (the development status of the tool)
- Representation (the techniques or technologies used to develop the tool)
- License (the type of license needed to use it)
- Publisher (the owner of the tool or the organization who is responsible for the publishing of the tool)
- Homepage (link to the documentation pages)

The current list of tools includes the following:

Global Sensor Networks - GSN

Description GSN is a Java environment that runs on one or more computers composing the backbone of the acquisition network. A set of wrappers allow to feed live data into the system. Then, the data streams are processed according to XML specification files. The system is built upon a concept of sensors (real sensors or virtual sensors, that is a new data source created from live data) that are connected together in order to built the required processing path. For example, one can imagine an anemometer that would sent its data into GSN through a wrapper (various wrappers are already available and writing new ones is quick), then that data stream could be sent to an averaging mote, the output of this mote could then be splited and sent for one part to a database for recording and to a web site for displaying the average measured wind in real time. All of this example could be done by editing only a few XML files in order to connect the various motes together.

Date of creation - last modification 11/11/2004 - now

Keywords data stream, sensor network, distributed system

Status UnderDevelopment

Representation Java, XML, XMLSchema

License GPL license, version 3

Publisher EPFL

Homepage `https://github.com/LSIR/gsn`

MonetDB

Description A relational database management system for high-performance data warehouses for business intelligence and eScience. Since a few years column store technology as pioneered in MonetDB has found its way into the product offerings of all major commercial database vendors. The market for applications empowered by these techniques provide ample space for further innovation, e.g. as demonstrated by our ongoing projects. At the same time, the landscape for major innovations remain wide open. A peek preview is given in the award winning paper titled: The Researcher's Guide to the Data Deluge: Querying a Scientific Database in Just a Few Seconds.

MonetDB is actively used in our research and real life applications. Nightly builds and regression testing ensure its quality, bug tracking helps to collect experiences and feature requests. Browsing the source code repository is supported by the Mercurial web frontend. Contributions ranging from bug reports, cross-platform issues, patches and features are highly appreciated.

Date of creation - last modification 2008 - now

Keywords column-store, XML, data management

Status UnderDevelopment

Representation Java, XML, XMLSchema

License MonetDB Public License

Publisher CWI

Homepage `http://www.monetdb.org/`

LDIF - Linked Data Integration Framework

Description The Web of Linked Data grows rapidly and contains data from a wide range of different domains, including life science data, geographic data, government data, library and media data, as well as cross-domain data sets such as DBpedia or Freebase. Linked Data applications that want to consume data from this global data space face the challenges that:

Data sources use a wide range of different RDF vocabularies to represent data about the same type of entity; The same real-world entity, for instance a person or a place, is identified with different URIs within different data sources; Data about the same real-world entity coming from different sources may contain conflicting value. For example the single value attribute population for a specific country can have multiple, different values after merging data from different sources. This usage of different vocabularies as well as the usage of URI aliases makes it very cumbersome for an application developer to write SPARQL queries against Web data which originates from multiple sources. In order to ease using Web data in the application context, it is thus advisable to translate data to a single target vocabulary (vocabulary mapping) and to replace URI aliases with a single target URI on the client side (identity resolution), before starting to ask SPARQL queries against the data.

Up-till-now, there have not been any integrated tools that help application developers with these tasks. With LDIF, we try to fill this gap and provide an open-source Linked Data Integration Framework that can be used by Linked Data applications to translate Web data and normalize URI while keeping track of data provenance.

Date of creation - last modification 29/6/2011 - 13/02/2014

Keywords linked data, data integration, schema mapping, identity resolution, data quality assessment, data fusion

Status Completed

Representation Java

License Apache

Publisher University of Mannheim and MES Semantics

Homepage `http://ldif.wbsg.de/`

D2RQ Plattform - Accessing Relational Databases as Virtual RDF Graphs

Description The D2RQ Platform is a system for accessing relational databases as virtual, read-only RDF graphs. It offers RDF-based access to the content of relational databases without having to replicate it into an RDF store. D2RQ allows: querying a non-RDF database using SPARQL, accessing the content of the database as Linked Data over the Web, creating custom dumps of the database in RDF formats for loading into an RDF store, and accessing information in a non-RDF database using the Apache Jena API D2RQ is Open Source software and published under the Apache license. The source code is available on GitHub.

Date of creation - last modification 08/12/2004 - 22/06/2012

Keywords Database-to-RDF Mapping, Linked Data Publication, SPARQL-to-SQL Rewriting

Status Completed

Representation Java, RDF, SQL

License Apache license

Publisher University of Mannheim and DERi Galway

Homepage `http://d2rq.org/`

linked-data-fu

Description Linked Data-Fu is a processing/scripting language for Linked Data based on (production) rules in Notation3 syntax. The rule engine evaluating Linked Data-Fu programs supports large-scale data integration (including reasoning with ontologies) and includes the data manipulation protocol (such as high-frequency updates). Linked Data-Fu allows for the writing of succint specifications that operate over a Read/Write Linked Data abstraction. Linked Data-Fu is an end-to-end data processing system that can be used in data integration and system interoperation scenarios.

Date of creation - last modification 08/09/2014-now

Keywords rdf

Status UnderDevelopment

Representation RDF

License Apache License v2

Publisher KIT

Homepage `http://aharth.github.io/linked-data-fu/`

morph-RDB

Description morph-RDB (formerly called ODEMapster) is an RDB2RDF engine developed by the Ontology Engineering Group, that follows the R2RML specification (http://www.w3.org/TR/r2rml/). Morph-RDB supports two operational modes: data upgrade (generating RDF instances from data in a relational database) and query translation (SPARQL to SQL). Morph-RDB employs various optimisation techniques in order to generate efficient SQL queries, such as self-join elimination and subquery elimination. Morph-RDB has been tested with real queries from various Spanish/EU projects and has proven to work faster than other state-of-the-art tools available. At the moment, Morph-RDB works with MySQL, PostgreSQL, and MonetDB.

Date of creation - last modification 2007 - now

Keywords rdb2rdf, r2rml, rdf, sql, rdb

Status UnderDevelopment

Representation Java, RDF, SPARQL

License Apache License v2

Publisher Universidad Politecnica de Madrid

Homepage `https://github.com/oeg-upm/morph-rdb`

morph-GFT

Description Morph-GFT is an extension of morph-RDB that works with Google Fusion Table (GFT) tables mapped with R2RML Mappings and enables users to query those tables using SPARQL queries. Underhood Morph-GFT, the SPARQL queries posed by the users are translated into SQL-like queries that are supported by GFT APIs. Unlike standard relational database implementation normally used with R2RML, GFT APIs do not support join operations. SPARQL-DQP is used to join the intermediate results and then the intermediate results are translated using the R2RML mappings specified by the users.

Date of creation - last modification 2007 - 12/11/2013

Keywords rdb2rdf, r2rml, rdf, sql

Status Completed

Representation Java, RDF, SPARQL

License Apache License v2

Publisher Universidad Politecnica de Madrid

Homepage `https://github.com/oeg-upm/morph-gft`

morph-LDP

Description The W3C Linked Data Platform (LDP) specification defines a standard HTTP-based protocol for read-/write Linked Data. The W3C R2RML recommendation defines a language to map relational databases (RDBs) and RDF. morph-LDP is a novel system that combines these two W3C standardization initiatives to expose relational data as read/write Linked Data for LDP-aware applications, whilst allowing legacy applications to continue using their relational databases.

Date of creation - last modification 2013 - now

Keywords rdb2rdf, r2rml, rdf, sql

Status UnderDevelopment

Representation Java, RDF, SPARQL

License Apache License v2

Publisher Universidad Politecnica de Madrid

Homepage `http://oeg-dev.dia.fi.upm.es/morph-ldp/`

morph-streams

Description SPARQL-Stream is a language that extends SPARQL for continuous query processing over streaming data. The Morph-streams module for SPARQL-Stream is a a java library that enables the execution of SPARQL-Stream queries, using different underlying DSMS or CEP (e.g. Esper, GSN, Cosm, SNEE, etc.). This tool allows posing SPARQL-Stream queries to an existing datasource using R2RML mappings. The mappings provide a descriptive way of relating ontological concepts (e.g. classes and properties) to elements of the DSMS or CERP schema (streams, tables). Morph uses a query rewriting approach to transform the SPARQL-Stream queries to native queries understandable and executable by the DSMS or CEP, using the R2RML mappings. Then, When morph executes the queries in the original datasources, it is capable of translating the responses to variable bindings or triples, depending on the type of query.

Date of creation - last modification 14/07/2011 - 12/02/2014

Keywords data stream, sensor network, query rewriting, SPARQL-Stream, query processing, RDF stream

Status Completed

Representation Scala

License GPL

Publisher UPM

Homepage `https://github.com/oeg-upm/morph-streams`

geometry2rdf

Description A tool that generates RDF triples from geometrical information, which can be available in GML or WKT. This will increase the possible reuse of that information. It converts the information into a format, RDF, that it's easier to consult and more reusable that the information available in the DDBB. geometry2rdf is a library for generating RDF files for geometrical information (which could be available in GML or WKT). The GML and WKT is manipulated with GeoTools. The current version of the library works with Oracle geospatial databases and relies on Jena.

Date of creation - last modification 01/02/2011 - 03/03/2013

Keywords geometry RDF transformation

Status Completed

Representation Java, RDF

License N/A

Publisher Universidad Politecnica de Madrid

Homepage `http://www.oeg-upm.net/index.php/technologies/151-geometry2rdf`

OOPS! - OntOlogy Pitfall Scanner!

Description OOPS! is a web-based tool, independent of any ontology development environment, for detecting potential pitfalls that could lead to modelling errors. This tool is intended to help ontology developers during the ontology validation activity, which can be divided into diagnosis and repair. Currently, OOPS! provides mechanisms to automatically detect a number of pitfalls, thus helps developers in the diagnosis activity.

Date of creation - last modification 14/11/2011 - now

Keywords ontology, ontology development, pitfall detection, ontology evaluation

Status UnderDevelopment

Representation Java

License GPLv3

Publisher Universidad Politecnica de Madrid

Homepage `http://www.oeg-upm.net/oops`

Videk

Description Videk is a mash-up based on several sources of data for environmental intelligence, including data coming from Smart Objects. Videk currently uses four sources of sensor and linked data and relies on StreamSense engine for storage and processing. On the server side Videk uses StreamSense, a sensor stream processing system based on tightly integrated and scalable custom software modules. StreamSense provides interfaces and means of information collection from a set of Smart Objects and generic APIs for data feeds on one hand; and interfaces to application developers on the other. Videk provides functionality such as, finding illuminance measurements around a given location or, showing all the locations in some region that measure illuminance.

Date of creation - last modification 2011 - now

Keywords Mash-up, sensors, web of things, real-time, data mining, semantic web.

Status UnderDevelopment

Representation XML

License N/A

Publisher JSI

Homepage `http://sensors.ijs.si/`

LODMiner

Description LOD Miner is a system for recommending missing properties for a given object. The input to the system is a set of objects or entities, each described with a set of properties. The system then tries to find the missing properties for a specified object based on similarity to other objects. Examples are RDF graphs from LOD.

Date of creation - last modification 2012 - now

Keywords linked open data, graph, missing properties, prediction.

Status UnderDevelopment

Representation Java

License N/A

Publisher JSI

Homepage `http://lodminer.net/`

IJS Newsfeed

Description Newsfeed provides a clean, continuous, real-time aggregated stream of semantically enriched news articles from RSS-enabled sites across the world.

The pipeline performs the following main steps:

1) Periodically crawl a list of RSS feeds and a subset of Google News and obtain links to news articles

2) Download the articles, taking care not to overload any of the hosting servers

3) Parse each article to obtain

3a) Potential new RSS sources mentioned in the HTML, to be used in step (1)

3b) Cleartext version of the article body

4) Process articles with Enrycher (English and Slovene only)

5) Expose two streams of news articles (cleartext and Enrycher-processed) to end users.

Date of creation - last modification 2012 - now

Keywords text, news, data stream, enrichment

Status UnderDevelopment

Representation XML

License LGPL

Publisher JSI

Homepage `http://newsfeed.ijs.si/`

CKAN Extractor - ckanext-extractor

Description CKAN plugin for the automatic extraction of data sources. It enables administrator to upload transformation plugins written in Python which extract data from non-structured sources. The extension provides a common framework for transformation development and periodic execution of tasks using celery.

Date of creation - last modification 27/02/2013 - 29/07/2013

Keywords data extraction

Status Completed

Representation Java

License AGPL

Publisher DeustoTech - Internet, MoreLab group

Homepage `https://github.com/morelab/ckanext-extractor`

CKANext-SILK

Description An extension for interlinking datasets uploaded on CKAN using SILK Link Discovery Framework. Using the declarative Silk - Link Specification Language (Silk-LSL), developers can specify which types of RDF links should be discovered between data sources as well as which conditions data items must fulfill in order to be interlinked. These link conditions may combine various similarity metrics and can take the graph around a data item into account, which is addressed using an RDF path language. Silk accesses the data sources that should be interlinked via the SPARQL protocol and can thus be used against local as well as remote SPARQL endpoints.

Date of creation - last modification 21/11/2012 - 11/03/2014

Keywords linked data, interlinking, CKAN, semantic web

Status Completed

Representation Java

License Apache License v2

Publisher DeustoTech - Internet, MoreLab group

Homepage `https://github.com/memaldi/ckanext-silk`

SPARQL Extension for CKAN - ckanext-sparql

Description This CKAN plugin offers two main functionalities: - It allows the configuration of a SPARQL endpoint for the whole CKAN platform in which metadata about every dataset in the platform can be queried. - It allows dataset editors to configure a RDF store and manage the RDF data of the dataset directly from CKAN, enabling at the same time a SPARQL endpoint for querying this data.

Date of creation - last modification 04/03/2013 - 11/03/2014

Keywords ckan, sparql, rdf store

Status Completed

Representation Java

License AGPL

Publisher DeustoTech - Internet, MoreLab group

Homepage `https://github.com/morelab/ckanext-sparql`

SPARQL endpoint analyzer and metadata generator for CKAN - ckanext-metadata

Description SPARQL endpoint analyzer and metadata generator for CKAN. It provides automatic extraction of remote SPARQL endpoints and calculates different properties such: number of subjects, number of predicates, properties, etc.

Date of creation - last modification 12/12/2012 - 11/03/2014

Keywords metadata, ckan, extension

Status Completed

Representation Java

License AGPL

Publisher UDeustoTech - Internet, MoreLab group

Homepage `https://github.com/morelab/ckanext-metadata`

Rhizomer

Description Rhizomer is a faceted browser that also provides a pivoting operation that allows no-experts to build complex semantic queries. It can be deployed on top of existing stores (Virtuoso, Jena, Sesame/OWLIM) and builds a user interface that provides Information Architecture components that facilitate fulfilling typical data exploration:

- Overview: global and local navigation menus are generated based on the classes instantiated by the dataset being published and also the SKOS concepts being subjects of the dataset resources.

- Zoom and Filter: when loading the dataset for the first time, it is analysed so it is possible to generate faceted views for all classes. Facets allow filtering using common values, searching for specific facet values and pivoting. The pivot operation allows switching from a particular faceted view, e.g. directors born in New Zealand, to faceted views of the sets related to the current resource set through one of the current facets, e.g. the faceted view of the films directed by the directors born in New Zaeland.

- Details on Demand: the RDF descriptions for the selected resources are rendered using an RDF2HTML+RDFa transformation. Moreover, it is also possible to use specialised visualisations like maps or timelines.

Date of creation - last modification 2008 - now

Keywords user interface, exploration, browser, Linked Data, Semantic Web, visualization

Status UnderDevelopment

Representation Java, XML, OWL, RDF, RDFSchema, SPARQL, SKOS

License GPL

Publisher Universitat de Lleida

Homepage `https://github.com/rhizomik/rhizomer`

HDT

Description HDT (Header, Dictionary, Triples) is a compact data structure and binary serialization format for RDF that keeps big datasets compressed to save space while maintaining search and browse operations without prior decompression. This makes it an ideal format for storing and sharing RDF datasets on the Web. Some facts about HDT:

- The size of the files is smaller than other RDF serialization formats. This means less bandwidth costs for the provider, but also less waiting time for the consumers to download.

- The HDT file is already indexed. The users of RDF dumps want to do something useful with the data. By using HDT, they download the file and start browsing/querying in minutes, instead of wasting time using parsing and indexing tools that are difficult to setup and tune.

- High performance querying. Usually the bottleneck of databases is slow disk access. The internal compression techniques of HDT allow that most part of the data (or even the whole dataset) can be kept in main memory, which is several orders of magnitude faster than disks.

- Highly concurrent. HDT is read-only, so it can dispatch many queries per second using multiple threads.

- The format is open and is acknowledged as W3C HDT Member Submission. This ensures that anyone on the Web can generate and consume files, or even write their own implementation.

- The libraries are open source (LGPL). You can adapt the libraries to your needs, and the community can spot and fix issues.

Date of creation - last modification 30/01/2013 - now

Keywords Binary RDF, compression, in-memory SPARQL, fast exchange.

Status UnderDevelopment

Representation Java, RDF, SPARQL, C++

License LGPL

Publisher DataWeb Research (University of Valladolid)

Homepage `http://www.rdfhdt.org/`

Yet Another SPARQL GUI

Description YASGUI is a web-based SPARQL client that can be used to query both remote and local endpoints. It integrates linked data services and web APIs to offer features such as auto-completion and endpoint lookup. It supports query retention - query texts persist across sessions - and query permalinks, as well as syntax checking and highlighting. Specially, YASGUI fits all requirements:

- Work on all endpoints (not just the CORS-enabled ones)
- Multi-platform (i.e. a web application)
- Easy-to-work user interface (i.e. prefix fetching, syntax highlighting/checking, storing queries)

Date of creation - last modification 01/07/2012 - now

Keywords SPARQL, Semantic Web, Endpoints

Status UnderDevelopment

Representation N/A

License MIT

Publisher

Homepage `http://laurensrietveld.nl/yasgui/`

Datalift Platform - Datalift
Description The Datalift web platform is a tool suite for converting, structured data sources and publishing them as linked data on the web. Datalift brings raw structured data coming from various formats (relational databases, CSV, XML, ...) to semantic data interlinked on the Web of Data. Datalift is an experimental research project funded by the French national research agency. Its goal is to develop a platform to publish and interlink datasets on the Web of data. Datalift will both publish datasets coming from a network of partners and data providers and propose a set of tools for easing the datasets publication process.
Date of creation - last modification 01/10/2010 - now
Keywords linked-data, structured data, interlinking, LOV, vocabulary mapping, ontologies, sql, shape, statistics, sdmx, datacube, CSV, SPARQL, JSON, Java, javascript, XML, RDF
Status UnderDevelopment
Representation Java, XML, OWL, RDF, RDFSchema, SPARQL, SKOS, SPIN
License Apache
Publisher INRIA
Homepage `http://datalift.org`

3.3 The PlanetData Tool Catalogue Organised According to Three Dimensions

We have categorized the collected tool descriptions according to three dimensions, following a set of categories that are not disjoint to each other (that is, a tool may belong to different categories under the same dimension):

1. By Input Data: includes three main categories—Stream Data, Linked Data, Non-Structured Data.
2. By Functionality: it consists of five categories—Produce, Publish, Consume, Provisioning, Data Management.
3. By Representation Technique: it considers the techniques or technologies used to develop the tool. This categorization contains 8 categories—Java, XML, RDF, SQL, SPARQL, C++, OWL, Scala.

3.3.1 Tools Catalogued By Input Data

In this section, we categorize the tools according to their target data formats. Since data is the main concern in PlanetData, it is important to know which tools should be used for specific types of datasets and dataset formats. Currently we use the following categories: Stream Data, Linked Data, Non-Structured Data that refers to the type of data which one can apply the given tool.

Table 3.1 depicts the tool catalogue with respect to the input data of each tool.

3.3.2 Tools Catalogued By Functionality

In this section, we categorize the collected tools according to their functionality. We use the following five categories:

- Produce: tools in this category allow generating new data.
- Publish: tools in this category are focused on publishing data (e.g., on web platforms).
- Consume: tools in this category are focused on processing data for specific applications; e.g. data mining, data compression.
- Provisioning: tools in this category are focused on manipulating data. This includes data conversion and data extraction tools, for instance.

Table 3.1: Tool Categories By Input Data

Name	Stream Data	Linked Data	Non-Structured Data
GSN	x		
MonetDB	x	x	
LDIF		x	
D2RQ		x	
linked-data-fu		x	
morph-rdb		x	
morph-gft		x	
morph-ldp		x	
morph-streams	x	x	
geometry2rdf		x	
OOPS!	x		
Videk	x	x	
LODMiner		x	
IJS Newsfeed			x
ckanext-silk		x	
ckanext-sparql		x	
ckanext-metadata		x	
ckanext-extractor			x
Rhizomer		x	
HDT		x	
Yet Another SPARQL GUI		x	
Datalift		x	

- Data Management: tools in this category allow managing data in large-scale scenarios; e.g. storage, indexing, and querying.

Table 3.2 illustrates the tool catalogue according to this dimension. Most of the tools are developed for consuming and provisioning data.

3.3.3 Tools Catalogued By Representation Technique

In this section, we categorize the tools according to their representation technique. It is important to know which technologies are used to develop a given tool. This helps users to know the compatibility between new and existing tools when they are integrated in the same platform. More precisely, we consider eight techniques: Java, XML, RDF, SQL, SPARQL, C++, OWL, Scala. Table 3.3 summarizes our tool catalogue according to the representation technique used for each tool.

Table 3.2: Tool Categories By Functionality

Name	Produce	Publish	Consume	Provisioning	Data Management
GSN	x	x	x		x
MonetDB			x		x
LDIF	x		x		
D2RQ			x	x	
linked-data-fu	x	x		x	
morph-rdb	x			x	
morph-gft	x			x	
morph-ldp	x			x	
morph-streams				x	
geometry2rdf	x			x	
OOPS!			x		
IJS Newsfeed		x			
Videk		x		x	
LODMiner			x		
ckanext-silk			x	x	
ckanext-sparql			x	x	
ckanext-metadata			x		
ckanext-extractor			x		
Rhizomer			x	x	
HDT			x		x
Yet Another SPARQL GUI			x	x	
Datalift		x		x	

Table 3.3: Tool Categories By Representation Technique

Name	Java	XML	RDF	SQL	SPARQL	C++	OWL	Scala
GSN	x	x	x			x		
MonetDB	x	x	x	x	x			
LDIF	x							
D2RQ	x		x	x				
linked-data-fu			x		x			
morph-rdb	x		x		x			x
morph-gft	x		x		x			
morph-ldp	x		x		x			x
morph-streams	x							x
geometry2rdf	x		x					
OOPS!	x		x				x	
IJS Newsfeed		x						
Videk		x						
LODMiner	x							
ckanext-silk	x		x					
ckanext-sparql	x		x		x			
ckanext-metadata	x		x					
ckanext-extractor	x		x					
Rhizomer	x	x	x	x	x	x	x	
HDT	x		x		x	x		
Yet Another SPARQL GUI	x							
Datalift	x	x	x		x	x	x	

4 Conclusions

In this deliverable, we have collected 22 tools, platforms and frameworks in total, all of them developed and maintained by PlanetData core and associated members. We have described them in a structured manner by using the ADMS vocabulary, which is the result of the JoinUp initiative. The RDF generated with the tool catalogue has been published on the PlanetData website, so as to allow for an easier browsing and querying of such metadata. Besides that, these tools, platforms and frameworks have been classified according to three main features, taking into account the following dimensions: by input data, by functionality, and by representation technique.