# The PLASMA Framework: Laying the Path to Domain-Specific Semantics in Dataspaces

Alexander Paulus
Institute for Technologies and Management of Digital Transformation
University of Wuppertal
Wuppertal, Germany
paulus@uni-wuppertal.de

André Pomp
Institute for Technologies and Management of Digital Transformation
University of Wuppertal
Wuppertal, Germany
pomp@uni-wuppertal.de

Tobias Meisen
Institute for Technologies and Management of Digital Transformation
University of Wuppertal
Wuppertal, Germany
meisen@uni-wuppertal.de

## ABSTRACT

Modern data management is evolving from centralized integration-based solutions to a non-integration-based process of finding, accessing and processing data, as observed within dataspaces. Common reference dataspace architectures assume that sources publish their own domain-specific schema. These schemas, also known as semantic models, can only be partially created automatically and require oversight and refinement by human modellers. Non-expert users, such as mechanical engineers or municipal workers, often have difficulty building models because they are faced with multiple ontologies, classes, and relations, and existing tools are not designed for non-expert users. The PLASMA framework consists of a platform and auxiliary services that focus on providing non-expert users with an accessible way to create and edit semantic models, combining automation approaches and support systems such as a recommendation engine. It also provides data conversion from raw data to RDF. In this paper we highlight the main features, like the modeling interface and the data conversion engine. We discuss how PLASMA as a tool is suitable for building semantic models by non-expert users in the context of dataspaces and show some applications where PLASMA has already been used in data management projects.

## CCS CONCEPTS

• **Information systems** → *Entity relationship models*; **Resource Description Framework (RDF)**; Web Ontology Language (OWL).

## KEYWORDS

semantic mapping, semantic model, graphical user interface, resource description framework, dataspace

**ACM Reference Format:**
Alexander Paulus, André Pomp, and Tobias Meisen. 2023. The PLASMA Framework: Laying the Path to Domain-Specific Semantics in Dataspaces. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3543873.3587662



*WWW '23 Companion, April 30–May 04, 2023, Austin, TX, USA*

ACM ISBN 978-1-4503-9419-2/23/04.
https://doi.org/10.1145/3543873.3587662

## 1 INTRODUCTION

Heterogeneous data management has risen to be one of the major directions in today's data exchange architectures. Data lakes and dataspaces [7] continue the shift from (structured) data integration towards on-demand data aggregation from multiple data sources in a pay-as-you-go fashion. In dataspaces, data is not transferred to a single location (as done in data lakes), but resides in the original location that provides the data. In theory, the dataspace does not impose any requirements towards that data regarding structure or guarantees and the data itself is managed solely by the source owner. Those data sources (often referred to as *Resources*) therefore provide highly heterogeneous data and might even change the structure or content of the data at any time or cease publishing data altogether. Due to the lack of a data integration which is controlled by a centralized entity, almost no guarantees can be made towards structure or format of the offered data. Still, in a dataspace, other participants expect to be able to obtain, process and extract data from the dataspace and therefore from those heterogeneous data sources.

While this approach offers several benefits over a pre-defined data integration using a fixed schema, like relational databases, finding, accessing and processing data from heterogeneous sources poses a major challenge. The International Data Spaces Reference Architecture Model (IDS RAM) [16], defined by the International Data Spaces Association (IDSA), specifies how data sources should be accessible in a dataspace. The corresponding reference specification is the IDS Information Model (IM) [1], which is an RDFS/OWL ontology defining fundamental concepts for describing actors and resources in dataspaces. It also provides building blocks for describing participant interactions, exchanged resources, as well as data usage restrictions. The upcoming IDS RAM 4.0[1] will expand on these concepts. Many dataspaces currently operational or in the making (e.g., Gaia-X [9], Catena-X[2] and others) use the IDS RAM as a blueprint to their dataspace architecture.

While existing reference specifications provide a foundation to describe data sources inside a dataspace, the data offered by those data sources remains largely undescribed[3] as the reference specification is intended to be independent of concrete application domains [1]. Still, the usage of RDF and domain ontologies is advocated as the preferred technology in order to provide domain

[1]https://docs.internationaldataspaces.org/ids-ram-4/
[2]https://catena-x.net/en/
[3]https://www.trusts-data.eu/data-spaces-semantic-interoperability/workshop-report-pictures-slides/

| title | directedBy | score | time | year |
|---|---|---|---|---|
| Interstellar | Christopher Nolan | Hans Zimmer | 169 | 2014 |
| Star Wars | George Lucas | John Williams | 121 | 1977 |
| Dune | Denis Villeneuve | Hans Zimmer | 155 | 2021 |

**Figure 1: Example dataset containing information about movies**

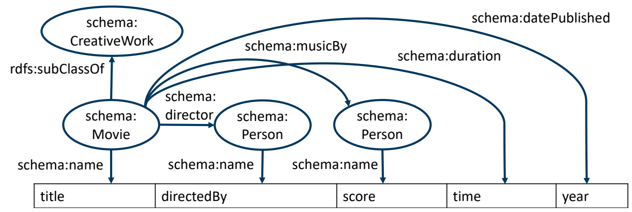


**Figure 2: Semantic model for the dataset shown in Figure 1**

specific semantics which detail the contents (i.e., datasets) of data sources.

Semantic models provide semantics to datasets in order to achieve a common understanding about the contents of that dataset [22]. They detail the single components (i.e., columns or fields) of a dataset while also providing context information to those components through the use of relations and meta concepts. As elements in semantic models are usually taken from shared conceptualizations like ontologies, semantic models provide a machine readable representation of the data contained. However, the necessary creation of semantic models poses two challenges. First, semantic models can rarely be fully generated in a completely automated fashion and often lack vital context information needed to properly interpret the contained data [18]. Second, having to provide a semantic model for a data source up front somehow violates the premise of dataspaces, which is to require no initial commitment from data providers. Assuming that domain experts often want to share data in a dataspace, these individuals are forced to create semantic models, a task that requires knowledge of semantic technologies that is rarely found among those workers.

In this paper, we first present the main features of the PLASMA framework, a tool to build and edit semantic models with the focus on non-expert users. The initial semantic modeling platform was presented in [19] and has since then been extended to include a data processing component which allows the conversion from raw data into RDF, using the semantic model as a template. In the following, we showcase how PLASMA assists non-expert users during the semantic model creation phase, positioning it as a tool suitable for the application in dataspaces.

The remainder of this paper is organized as follows. We briefly introduce the basics of semantic models and detail the semantic model creation process in Section 2. PLASMA and its components are then described in Section 3. We present three fields of application where PLASMA has been used in productive environments with non-expert users in Section 4. Related work is summarized in Section 5. We conclude with a summary and an outlook towards future developments in Section 6.

## 2 SEMANTIC MODELS

We first give a short introduction into the nature, shape and purpose of semantic models. In Section 2.2, a closer look will be taken at the process of creating a semantic model and which pitfalls might pose challenges to inexperienced users during their model creation.

### 2.1 Data Interpretation and Semantic Models

Relational databases define a name and data type for each column in their tables. Although the data type reveals nothing about the content of the data, most field names are often helpful in understanding the data. Those field names (and relations to other fields or tables) provide context to the raw data, helping data scientists or software developers to interpret the values stored in the field or column. When it comes to dataspaces or open data portals, data is usually shared in an at least semi-structured form, e.g. as a CSV or JSON file, that provides similar context as a database schema. Figure 1 shows an example dataset containing information about movies, their director, composer, run time and publication year which might be provided as a CSV file. Without any context information, some deductions towards the contents of each column can be done by a human observer: first column identifies the movie title, second column the director. The name of the third column could either reference to the title of the accompanying soundtrack or the composer. From the examples values contained in the column one might identify the composer as the more natural choice as the contents refer to names of persons. While the forth column obviously contains a duration (deductible from the header and the values) it is not immediately clear if the contained values are in minutes or seconds. Only by regarding the context of that column, an observer might correctly identify the unit of measurements as minutes. However, even with this information, it is unclear if this measure refers to the movie or the soundtrack run time, as both appear to be suitable choices. The same conditions apply for the "year" column, which at first has to be identified as a year of release but could also refer to both the soundtrack and the movie itself.

Human observers might be able to deduct the meaning of all those columns based on context information from other columns, possible known relationships between different combinations of candidate meanings, experience and intuition. Most of those techniques are unavailable to computers and even if available (e.g. through the use of machine learning), would still not result in a fully confident interpretation of the values (cf. Section 2.2). Applying those interpretation processes to a new dataset obtained from an unknown machine with partially cryptic column names will likely fail for both human and machines as context, intuition and experience are missing.

In order to provide those necessary information, semantic technologies can be used. A semantic model is a formalization of the semantic information needed to interpret the data contained in the dataset. Therefore, a proper semantic model contains all information, expressed as concepts and relations, needed to interpret the data correctly. Figure 2 shows the semantic model for the dataset presented in Figure 1, depicted as a graph.

In a semantic model, nodes represent concepts (classes) while edges represent relations (predicates/properties) between those classes or the data fields. Concepts and relations are usually obtained from ontologies (e.g. schema.org[4]). Semantic models are coded using the RDF triple format (subject, predicate, object) [11], with each relation being expressed as a triple, e.g., (schema:Movie, schema:musicBy, schema:Person). As RDF is machine readable, semantic models empower machines to be aware of the internal semantics of the dataset, similar to human observers, but also to index and query the data using the semantic model as schema information for a structured query (e.g. SPARQL [11]).

## 2.2 Model Creation

In recent years, automation in the field of semantic model creation has seen some significant advances [18], while new techniques continue to improve different aspects of fully automated semantic model generation. The major focus in this field lies on the *semantic labeling* step, which assigns initial concepts to fields of a database. From this initial assumption, the advanced *semantic modeling* identifies additional relations and meta concepts to build an initial semantic model.

However, various authors of state-of-the-art automation algorithms still identify a need for manual refinement or creation of the generated models [8, 19, 23]. Since most modern automation approaches are substantially based on machine learning, the quality of predictions depends on the quality and quantity of training data. In domains where little or no historical data is available (cold start problem), automation fails to provide adequate results, requiring human modelers to step into the process. Additionally, in the context of dataspaces where semantic models are supposed to interpret the values contained, some parts of semantic models can never be properly filled using automated approaches as those are dependent on the data collection environment [18]. Examples where manual editing is necessary include units of measurement, reference values (e.g., applied offsets) as well as locations or any other information necessary for interpretation that is not contained in the actual data (e.g., an identifier of the machine from which the data was collected). This information must be entered into the model by human supervisors, requiring engineers, municipal workers or other data providers to edit the semantic model after the initial automated generation.

Editing semantic models poses some challenges to non-expert users. Available tools, such as SAND [26] or the RML Editor [10] require in-depth knowledge of RDF, ontologies and the mapping process, making these tools unsuitable for the target users (cf. Section 5). Additionally, suitable ontologies, which provide the foundation of semantic models, are often missing for specific application domains, resulting in users being unable to model certain facts or requiring them to combine concepts and relations from various ontologies.

# 3 THE PLASMA FRAMEWORK

To overcome these issues, we introduced PLASMA (PLatform for Auxiliary Semantic Modeling Approaches) as a modular framework to serve as a baseline semantic modeling tool [19]. The main component of PLASMA is the modeling web based UI paired with the Data Modeling Service (DMS) which stores the models and editing history. Finished models are stored in the Knowledge Graph Service (KGS) which manages an internal knowledge graph as well as ontologies imported into PLASMA. An additional optional component, the Data Processing Service (DPS), can be used to convert raw JSON data into RDF using a semantic model obtained from the DMS. The platform architecture is realized as a collection of microservices, which allows PLASMA to be used as either a standalone modeling tool or embedded into a larger software construct (see Section 4).

[4]https://schema.org

## 3.1 Semantic Model Creation in PLASMA

PLASMA provides a graphical user interface (Figure 3) to display the current state of the semantic model as a two-layered graph. The background layer displays the *syntax model* which resembles the structure of the input data obtained through a schema analysis (low opacity elements like the ROOT node in Figure 3). The input of, e.g., a CSV file is rendered in a simple tree structure with an artificial ROOT node. In addition to tabular data, PLASMA is also capable of handling hierarchical data such as JSON or XML files which are displayed in a tiered way to organize nested elements. The overall structure of the syntax model aims to provide an initial view that is similar to the data structure the modeler is familiar with. The graph's foreground layer displays the current state of the semantic model. This layer expands with each element the user adds to the semantic model and is rendered on top of the lower layer. The visualization resembles a graph representation of RDF triples. Nodes and edges indicate subjects/objects (classes and literals) and edges, respectively. By default, most RDF-specific details are hidden for simplicity. The graph uses defined labels (if available) for classes and properties instead of URIs. A color coding helps distinguishing classes from literals. Users may freely arrange elements in both syntax and semantic representation layers, e.g., to cluster nodes that represent data related to each other. A more detailed description of the modeling UI can be found in [17]. The majority of semantic model creation in PLASMA is done using drag & drop interactions. Creating a semantic model is achieved by dragging classes from an extensible drawer on the left and dropping them into the modeling area. Dropping a class onto the canvas creates a class used for meta information. In case a class is dropped onto a node in the syntax model, a mapping is created between that node and the class if the node contains data (leaf node). Relations can be drawn by selecting the desired item in the relations tab and dragging a line from one node to another. All available elements (classes and relations) are taken from a set of ontologies managed in PLASMA. The user may filter which ontologies to search but also upload new ontologies to PLASMA. Currently, only OWL ontologies are supported. Ontologies in other formats (like schema.org) may be converted into the OWL format to be used in PLASMA. If no matching class or relation can be found in an ontology previously imported into the system, PLASMA offers users the ability to create a *provisional* element manually during the semantic model editing using a local ontology managed by the KGS. This principle is inspired by the idea of Lipp et al. [14] to allow domain experts to quickly create ontologies. Each element can be defined by specifying a type (class or named entity and OWL ObjectProperty/DatatypeProperty), label, description and URI and added into a cache visible only to the current modeling process. The modeler can then use that element

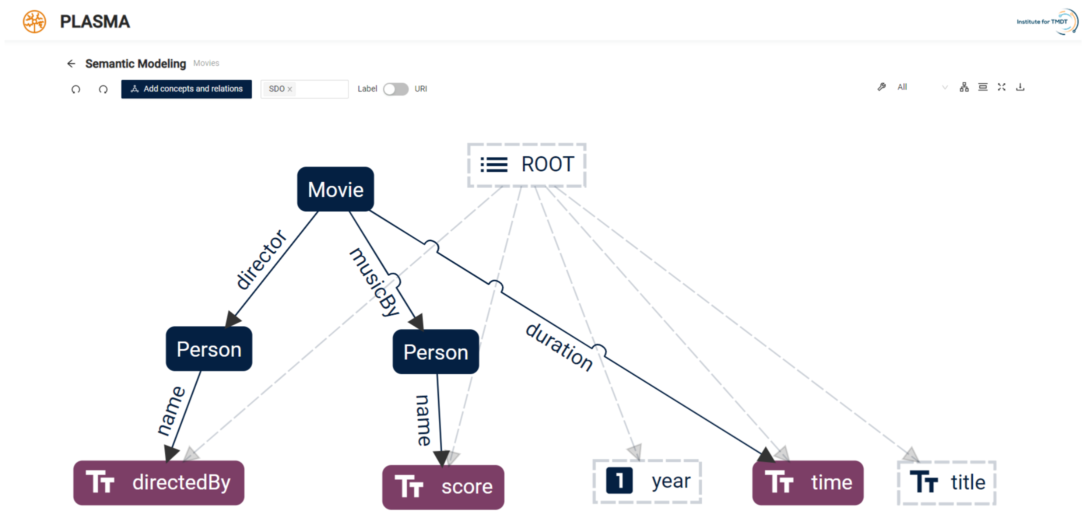


**Figure 3: Screenshot of the PLASMA modeling interface**

during the modeling similar to elements obtained from imported ontologies. It is also possible to edit the element later on during the modeling. The information gathered in the local ontology may be used later on to build a new, domain specific ontology [14]. At any time during the modeling, the current model can be exported as an RDF graph. A video screencast showing an example basic modeling process is available online[5].

*3.1.1 Assistance Technologies and Tools.* While the modeling interface is still the main focus, the PLASMA architecture allows to add components that provide assistance technologies. Those technologies have been added to speed up the modeling process and to provide some guidance during manual modeling. Assistance technologies include automation algorithms, e.g., for automated semantic labeling and modeling, as well as recommendations for the user to accept or reject. The underlying software components are called Auxiliary Recommendation Services (ARS) and can be consulted at any stage during the modeling phase. This allows the user to, for example, first request a semantic labeling, correct any existing errors and only then let an ARS generate an initial semantic model, reducing error propagation between those two steps. In case the computed result is undesired, PLASMA keeps track of the whole editing history and provides an *undo* function to return to previous modeling states.

Alongside the ARS for automation, PLASMA also features the display of recommendations which are to be created by an ARS. Recommendations are displayed as highlighted nodes in the actual semantic model and contain possible additions to the semantic model. This way, missing context information might be added by the user through a recommendation to add this specific fact to the model. Recommendations may also reduce the time to identify single elements matching elements from the ontologies, saving valuable modeling time. A single ARS for recommendations based on statistical information obtained from all models in the PLASMA instance is already available to provide a baseline for recommendation generation. In the future, we plan to integrate multiple other ARS into PLASMA, featuring recent approaches in the domains of semantic labeling (e.g. [4, 12, 20]), modeling [8, 27] and upcoming recommendation algorithms. We also encourage the community to add their algorithms to PLASMA as ARS.

In addition to the ARS, the PLASMA modeling interface has undergone many other enhancements over the years to meet the changing needs of users and support them with additional assistance technologies. For example, in newer versions it is possible to copy a whole semantic model from another modeling process. After defining the new mappings and optionally editing the model, it can be exported, saving a majority of time in case multiple similar datasets are to be modeled.

[5]https://bit.ly/3OmeJlE

## 3.2 RDF Data Conversion

```
plsm:4b1efca7-2681-2  rdf:type  schema:Person ;
        schema:name  "Hans Zimmer".

plsm:4b1efca7-2681-1  rdf:type  schema:Person ;
        schema:name  "Christopher Nolan".

plsm:4b1efca7-2681-0  rdf:type  schema:Movie ;
        schema:datePublished  "2014";
        schema:director       plsm:4b1efca7-2681-1 ;
        schema:duration    "169";
        schema:musicBy        plsm:4b1efca7-2681-2 ;
        schema:name        "Interstellar".

plsm:4b1efca7-2681-5  rdf:type  schema:Person ;
        schema:name  "George Lucas".
```

**Listing 1: Turtle respresentation of data conversion result (partial)**

In addition to creating semantic models, PLASMA is also capable of converting raw data to RDF. PLASMA's central internal data structure is called the *Combined Model* (CM), which contains all information about the contents of both syntax and semantic models, existing mappings between those models, and provisional elements (see Section 3.1). From the CM, the current semantic model for a selected input file is used as a blueprint for the RDF conversion. An engine parses the semantic model node by node and either creates an instance for each context class or fills a literal with either the respective data from the currently converted data point or a static value defined in the semantic model. Afterwards, all relations are copied from the CM to the generated RDF model. If, for example, the data shown in Figure 1 is converted using the semantic model shown in Figure 2, the resulting RDF output will resemble the triples in Listing 1. Identifiers for output nodes are generated based on the (random) model id. The PLASMA conversion engine roughly behaves like the RMLMapper[6], but adds some specific features requested by previous users. As an example, in geospatial datasets, polygons are modeled as lists of points. Those can be mapped in RML too, but the necessary order is lost in the mapping step. PLASMA models the contained data using RDF lists, thus preserving order and also allowing relations to be created between other instances and single items in the list as well as the list itself.

### 3.3 Setup

All components of PLASMA are available from GitHub[7] under an MIT license. The backend components are written in Java while the frontend is based on Angular. Due to the microservice architecture, other languages like Python may be used for the (mostly machine learning based) ARS. Execution is based on Docker images for each microservice. The deployment can be achieved by building the Docker containers locally using Maven and then using the provided compose script to start up PLASMA. A demo instance for testing is available at http://plasma.uni-wuppertal.de.

## 4 APPLICATIONS

As mentioned in Section 3, PLASMA can be used either as a standalone modeling tool or integrated into other software applications. In the following, we outline application scenarios in which PLASMA is currently operated.

### 4.1 PLASMA as Standalone Tool

An instance of PLASMA is operated at a German conglomerate with focus on automation and digitization to create semantic models for time series data [17]. In this application, the structure of a data set, such as, a time series of measurements, is extracted and PLASMA is used by domain experts to create a semantic model using predefined ontologies, such as SOSA. Once the modeling creation is complete, the exported RDF model can be used to process the data, such as integrating it into a knowledge base such as an industrial knowledge graph.

In the second application, PLASMA is currently being evaluated by the city of Wuppertal for the creation of 5-star open data [17]. Here, domain experts from the Geodata Management Department are creating semantic models for their open data. PLASMA's data conversion functionality is then used to generate RDF data, to be published on the open data portal. To train the different staff members, a workshop was held to teach the basics of semantic modeling.

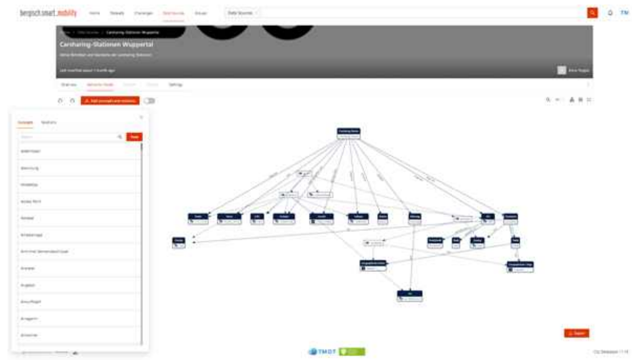

**Figure 4: PLASMA in the City Dataspace**

### 4.2 PLASMA in the City Dataspace

The third application uses PLASMA as an integrated component within a dataspace. The City Dataspace [21] aims to enable the data-driven smart city of tomorrow by enabling the compatibility of heterogeneous open and urban data from different municipalities, combined with data from other stakeholders across city boundaries. To this end, the City Dataspace relies on the established semantic data management mechanisms. Datasets added to the dataspace require the definition of a semantic model in order to improve the data quality towards defined FAIR goals. When extracting data, the user can choose which parts of the data to export based on the semantic model, using the semantic descriptions as a guide. To date, more than 200 data sources have been created by eight municipal domain experts in the field of geospatial information, who created the models without in-depth knowledge of semantic technologies. Figure 4 shows the user interface of PLASMA within the City Dataspace. A corpus of semantic models created using PLASMA has been published as the VC-SLAM dataset [5].

In a dataspace, PLASMA's data converter can be used to provide data contained in a data source in a standardized format (i.e., RDF) that exactly matches the semantic structure observed in the semantic model. This allows data consumers to query the data from the data source using a query language like SPARQL, strengthening interoperability between different data sources as the data may be processed inside a single query across multiple data sources. Alternatively, data has to be queried in a raw format, collected at the consumer and then manually processed afterwards, as multiple data schemes have to be aligned.

## 5 RELATED WORK

Increasing the accessibility of semantic technologies is not a task that is unique to the field of semantic modeling. This approach is already being followed, especially in the field of ontology engineering. A sizable number of ontology editors have been developed over the years, such as approaches presented in [3, 6, 15, 24]. However, their complexity makes them unsuitable for non-expert users that do not have any or only little prior knowledge in semantic technologies. In the last two years, several new approaches have been published that

[6]https://github.com/RMLio/rmlmapper-java
[7]https://github.com/tmdt-buw/plasma

feature convenient visual interfaces, highlighting the importance of proper visualization [2, 14, 25]. However, as ontology editors, they are not capable of handling data and including mappings, thus rendering them unusable for semantic model creation.

One of the first semantic model creation tools was Karma [13] which already visualized multiple different generated semantic models for the user to pick from. Its editing and building capabilities are however limited, narrowing the field of application to inspection only. SAND [26], a semantic modeling tool that provides a visualization of the model alongside a table view, allows creating semantic models and offers the ability to convert the raw data to RDF or JSON-LD. The tool is limited to tabular data whereas PLASMA is able to convert hierarchical data too. The RML Editor [10] supports mapping hierarchical data, but is limited in the complexity of created semantic models as some relation combinations are not supported by the underlying RML mapping engine.

## 6 CONCLUSION AND FUTURE WORK

In this paper, we presented how PLASMA, a semantic model creation tool featuring a visual editing display, could be used in a dataspace environment. PLASMA is capable of (1) creating semantic models in an environment convenient for non-expert users, (2) provide continuous support during the model refinement through a modular assistance framework, (3) maintaining an internal knowledge graph and extendable ontology, (4) export semantic models to provide a semantic description for datasets and (5) convert data into RDF to provide a unified format for data exchange. We highlighted three scenarios in which PLASMA has been used so far to generate semantic models and optionally convert data into standardized RDF. While other tools partially provide similar functionality, none of the existing tools offer all the benefits of PLASMA in one modeling environment.

For future work, we would like to extend the initial set of ARS to include more support functions into PLASMA. Furthermore, we plan to include an export of the mapping as an R2RML configuration to increase compatibility to existing systems. In addition, a special modeling and export mode for JSON-LD is planned which adds semantic information to the original JSON structure instead of converting the data into RDF. This will increase usability of the data to consumers that require adherence to the original data format, which are often encountered in dataspaces.

## REFERENCES


[1] S. Bader, J. Pullmann, C. Mader, S. Tramp, C. Quix, et al. 2020. The International Data Spaces Information Model – An Ontology for Sovereign Exchange of Digital Content. In *The Semantic Web – ISWC 2020*. Springer International Publishing.

[2] Shadi Baghernezhad-Tabasi, Loïc Druette, Fabrice Jouanot, Celine Meurger, and Marie-Christine Rousset. 2021. IOPE: Interactive Ontology Population and Enrichment. In *Poster&Demo track and Workshop on Ontology-Driven Conceptual Modelling of Digital Twins co-located with Semantics 2021*. Amsterdam, Netherlands. https://hal.archives-ouvertes.fr/hal-03448159

[3] Sean Bechhofer, Ian Horrocks, Carole Goble, and Robert Stevens. 2001. OilEd: a reason-able ontology editor for the semantic web. In *Annual Conference on Artificial Intelligence*. Springer, 396–408.

[4] Andreas Burgdorf, Alexander Paulus, André Pomp, and Tobias Meisen. 2022. DocSemMap 2.0: Semantic Labeling Based on Textual Data Documentations Using Seq2Seq Context Learner. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 98–107. https://doi.org/10.1145/3511808.3557446

[5] Andreas Burgdorf, Alexander Paulus, André Pomp, and Tobias Meisen. 2022. VC-SLAM - A Handcrafted Data Corpus for the Construction of Semantic Models. *Data* 7, 2 (2022). https://doi.org/10.3390/data7020017

[6] Blaz Fortuna, Marko Grobelnik, and Dunja Mladenic. 2007. Ontogen: Semi-automatic ontology editor. In *Symposium on Human Interface and the Management of Information*. Springer, 309–318.

[7] M. Franklin, A. Halevy, and D. Maier. 2005. From databases to dataspaces: a new abstraction for information management. *ACM Sigmod Record* 34, 4 (2005).

[8] Giuseppe Futia, Antonio Vetrò, and Juan Carlos de Martin. 2020. SeMi: A SEmantic Modeling machIne to build Knowledge Graphs with graph neural networks. *SoftwareX* 12 (2020), 100516. https://doi.org/10.1016/j.softx.2020.100516

[9] GAIA-X European Association for Data and Cloud. 2022. *Gaia-X Architecture Document*. Technical Report. Gaia-X European Association for Data and Cloud AISBL, Avenue des Arts 6-9, 1210 Brussels. Retrieved January 29, 2023 from https://docs.gaia-x.eu/technical-committee/architecture-document/22.10/

[10] Pieter Heyvaert, Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, Erik Mannens, and Rik Van de Walle. 2016. RMLEditor: A Graph-Based Mapping Editor for Linked Data Mappings. In *The Semantic Web. Latest Advances and New Domains*, Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange (Eds.). Springer International Publishing, Cham, 709–723.

[11] Aidan. Hogan. 2020. *The Web of Data* (1st ed. 2020 ed.). Springer International Publishing and Imprint: Springer, Cham.

[12] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çağatay Demiralp, and César Hidalgo. 2019. Sherlock: A Deep Learning Approach to Semantic Data Type Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

[13] Craig A. Knoblock, Pedro Szekely, et al. 2012. Semi-Automatically Mapping Structured Sources Into the Semantic Web. In *Extended Semantic Web Conference*.

[14] Johannes Lipp, Lars Gleim, Michael Cochez, Iraklis Dimitriadis, Hussain Ali, Daniel Hoppe Alvarez, Christoph Lange, and Stefan Decker. 2021. Towards Easy Vocabulary Drafts with Neologism 2.0. In *The Semantic Web: ESWC 2021 Satellite Events*, Ruben Verborgh, Anastasia Dimou, Aidan Hogan, Claudia d'Amato, Ilaria Tiddi, Arne Bröring, Simon Mayer, Femke Ongenae, Riccardo Tommasini, and Mehwish Alam (Eds.). Springer International Publishing, Cham, 21–26.

[15] Mark A. Musen. 2015. The protégé project: a look back and a look forward. *AI Matters* 1, 4 (2015), 4–12. https://doi.org/10.1145/2757001.2757003

[16] B. Otto, S. Steinbuss, A. Teuscher, and S. Lohmann. 2019. IDS Reference Architecture Model. https://doi.org/10.5281/ZENODO.5105529

[17] Alexander Paulus, Andreas Burgdorf, Tristan Langer, André Pomp, Tobias Meisen, and Sebastian Pol. 2022. PLASMA: A Semantic Modeling Tool for Domain Experts. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 4946–4950. https://doi.org/10.1145/3511808.3557184

[18] Alexander Paulus, Andreas Burgdorf, André Pomp, and Tobias Meisen. 2021. Recent Advances and Future Challenges of Semantic Modeling. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*. 70–75. https://doi.org/10.1109/ICSC50631.2021.00016

[19] Alexander Paulus., Andreas Burgdorf., Lars Puleikis., Tristan Langer., André Pomp., and Tobias Meisen. 2021. PLASMA: Platform for Auxiliary Semantic Modeling Approaches. In *Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 2: ICEIS,*. INSTICC, SciTePress, 403–412. https://doi.org/10.5220/0010499604030412

[20] Minh Pham, Suresh Alse, et al. 2016. Semantic Labeling: A Domain-Independent Approach. In *The Semantic Web – ISWC 2016*. Springer International Publishing.

[21] André Pomp, Alexander Paulus, Andreas Burgdorf, and Tobias Meisen. 2021. A Semantic Data Marketplace for Easy Data Sharing within a Smart City. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 4774–4778.

[22] Mohsen Taheriyan, Craig A. Knoblock, et al. 2014. A Scalable Approach to Learn Semantic Models of Structured Sources. In *2014 IEEE International Conference on Semantic Computing (ICSC)*. IEEE, Piscataway, NJ, 183–190.

[23] Mohsen Taheriyan, Craig A. Knoblock, et al. 2016. Learning the Semantics of Structured Data Sources. *Journal of Web Semantics* 37-38 (2016), 152–169.

[24] Tania Tudorache, Jennifer Vendetti, and Natalya Fridman Noy. 2008. Web-Protege: A Lightweight OWL Ontology Editor for the Web.. In *OWLED*, Vol. 432. 2009.

[25] Andre Valdestilhas, Gustavo Publio, Andrea Cimmino Arriaga, and Thomas Riechert. 2021. VocEditor-An Integrated Environment to Visually Edit, Validate and Versioning RDF Vocabularies. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*. IEEE, 473–476.

[26] Binh Vu and Craig A. Knoblock. 2022. SAND: A Tool for Creating Semantic Descriptions of Tabular Sources. In *The semantic web*, Paul Groth (Ed.). Lecture Notes in Computer Science, Vol. 13384. Springer, Cham, 63–67.

[27] Jiakang Xu, Wolfgang Mayer, Hongyu Zhang, Keqing He, and Zaiwen Feng. 2022. Automatic Semantic Modeling for Structural Data Source with the Prior Knowledge from Knowledge Base. *Mathematics* 10, 24 (2022). https://doi.org/10.3390/math10244778