# Composable Semantic Data Transformation Pipelines with Chimera

Marco Grassi, Mario Scrocca, Alessio Carenini, Marco Comerio and Irene Celino

*Cefriel – Politecnico di Milano, Milan, Italy*

**Abstract**

In a multi-stakeholder ecosystem, data exchange is not sufficient and semantic interoperability should be achieved to ensure different information systems can communicate without loss of meaning. A semantic conversion procedure relying on a reference ontology can enable an efficient conversion between data formats sharing common semantics, whilst generating an integrated and interoperable knowledge graph. The Chimera framework proposes a flexible and configurable solution to address different requirements for the integration of semantic converters across heterogeneous systems. Chimera minimises the effort required to specify custom semantic data transformation pipelines, offering different ready-to-use components to manipulate knowledge graphs and execute declarative mapping rules for their construction and exploitation. We present the Chimera resource and we demonstrate its usage considering a conversion use case in the transportation domain.

**Keywords**

Semantic conversion, Declarative mappings, Data interoperability

## 1. Introduction

The issue of data interoperability is a significant concern when operating within a multi-stakeholder ecosystem where diverse actors employ heterogeneous data formats, specifications, and semantics. The ability to exchange data without any loss of meaning among communicating parties is an essential objective, but it is notoriously challenging to achieve also due to different requirements for the integration of heterogeneous information systems. Chimera[1], is an open-source framework for the definition of composable semantic data transformation pipelines. The proposed solution addresses data interoperability issues by leveraging Semantic Web technologies to extract and harmonise the intended semantics of heterogeneous data sources by means of transformation procedures. The framework is designed as an extendable set of building blocks to simplify the configuration of semantic data transformations and to provide flexibility in addressing diverse scenarios and requirements. Chimera, originally designed to support the use case presented in [1], has been completely refactored to address more generic requirements and to facilitate its extensibility and reusability. In this paper, we discuss the

*KGC '23: Knowledge Graph Construction Workshop co-located with the ESWC 2023, May 28, 2023, Hersonissos, Greece*
marco.grassi@cefriel.com (M. Grassi); mario.scrocca@cefriel.com (M. Scrocca); alessio.carenini@cefriel.com (A. Carenini); marco.comerio@cefriel.com (M. Comerio); irene.celino@cefriel.com (I. Celino)
000-0003-3139-3049 (M. Grassi); 0000-0002-8235-7331 (M. Scrocca); 0000-0003-1948-807X (A. Carenini); 0000-0003-3494-9516 (M. Comerio); 0000-0001-9962-7193 (I. Celino)
© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

[1]Chimerav3,https://github.com/cefriel/chimera

main concepts behind Chimera, how it can be reused for different use cases, and its application to support a complex scenario in the transportation domain. The remainder of the paper is organised as follows: Section 2 provides an analysis of the challenges associated with the semantic data transformation process, Section 3 presents the Chimera framework, Section 4 exemplifies the usage of Chimera and describes the resources available, Section 5 compares Chimera with existing alternative solutions, Section 6 presents conclusions and future work.

## 2. Problem description

In multi-stakeholder systems, different actors may adopt diverse data standards, which can result in data fragmentation and incompatibility. To effectively manage and utilize data in such complex environments, data integration and interoperability become essential. Data integration involves the process of merging and consolidating data from disparate sources to create a unified view, while data interoperability refers to the ability of different systems and applications to communicate and exchange data seamlessly. By ensuring these two aspects, stakeholders can better collaborate and make informed decisions based on accurate and comprehensive data. Moreover, the definition of additional services is enabled by relying on integrated and harmonised data from different information systems.

The commonly arising challenge is about how to deal with heterogeneous data formats with varying semantic interpretations employed by multiple actors within the same domain. This phenomenon may arise due to several factors that make the establishment of standards difficult, for example, the persistence of legacy applications or the usage of proprietary data formats. This scenario is usually addressed through the definition of custom solutions (e.g., ad-hoc software components) that are able to interconnect different information systems and address the conversion of data between their respective data formats. However, these solutions are specifically crafted and are neither easily maintainable nor scalable. The challenge addressed by Chimera is twofold and concerns how to support data interoperability in a multi-stakeholder ecosystem relying on different data formats (Challenge 1) and heterogeneous information systems with different functional requirements for data exchange (Challenge 2).

The adoption of an any-to-one centralized mapping approach [2] relying on a global conceptual model offers a scalable approach to address the first challenge. Indeed, each stakeholder only needs to define mappings to/from the global conceptual model and the number of mappings required for conversion and compliance among different data formats is heavily reduced. If there are $n$ different formats, the number of mappings is $2n$ instead of $2n(n-1)$. The implementation of such an approach by relying on Semantic Web technologies provides further advantages [1]. The global conceptual model can be identified through a reference ontology, that represents the entities and relationships of the specific domain considered, and bilateral mappings can be defined between specific formats and the reference ontology. On one hand, the two-step semantic conversion process (i.e., lifting from the original format to the reference ontology and then lowering from the ontological version to the target format) has a positive "collateral effect" of building a knowledge graph of interlinked and interoperable information (cf. 1). On the other hand, the usage of mappings solution relying on a declarative approach facilitates the maintainability and reproducibility of the conversion process [3].

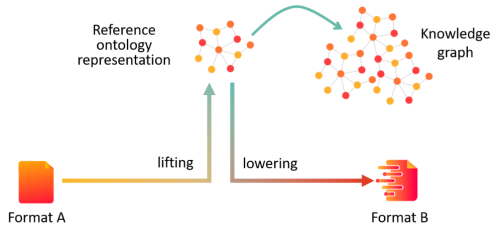


**Figure 1:** Building a knowledge graph as a "collateral effect" of data conversion.

The second challenge deals with the fact that there is no single interoperability problem and, therefore, it is not possible to define a single interoperability solution [4]. A flexible and configurable set of specialized tools is needed to cope with different functional requirements to implement data interoperability across heterogeneous information systems. While a semantic conversion process can offer a valid solution to define transformations across data formats, the integration of such processes considering different data sources and sinks is something that requires a case-by-case analysis. The two main use cases to be considered are the *batch conversion* and the *service mediation* use cases [5].

*Batch conversion* refers to the case where a possibly large set of data should be converted and the conversion process needs to be run sporadically, e.g., only when the original dataset changes. This case usually considers the conversion of data directly extracted from databases and/or of datasets stored on the cloud (e.g., using Object Storage services). In this case, the requirement is to limit the usage of resources during the conversion process.

*Service mediation* considers the conversion as an intermediate step to enable data exchange between stakeholders, usually in real time. In this case, the conversion process should be continuously available and integrated with different communication interfaces used to emit and receive data, e.g., HTTP API, WebSocket, MQTT. This data may have time-related constraints and requirements in terms of concurrent requests to be served, moreover, usually low overhead is tolerated.

Both use cases should be then further declined considering protocols and technologies adopted by the involved systems and the specific requirements of the addressed scenario. As an example, we consider in this paper a use case emerging from the European H2020 TANGENT project[2] in which Chimera is used to define a solution for data harmonisation and fusion in the context of multimodal traffic management. In this scenario, the objective is to implement a real-time integrated dashboard for city and transport authorities that could enable intelligent incident detection and thus facilitate prompt intervention. To this end, real-time data from multiple data sources such as semaphores, parking sensors and traffic sensors need to be collected and integrated. Additionally, data related to planned events, such as concerts or sporting events need to be taken into account because they have a large and predictable effect on traffic conditions. The definition of a semantic conversion process should consider the following requirements: data from sensors and devices are collected through a Kafka[3] deployment but are represented

[2]https://tangent-h2020.eu/
[3]https://kafka.apache.org/

using heterogeneous data formats, planned events are stored as datasets in a Data Catalogue, integrated and harmonised data should be stored (e.g., for the training of machine learning models on historical data), a monitoring platform is used to track relevant events about the process, a specific JSON format is expected by the dashboard that receives data through a WebSocket.

The presented scenario highlights the need for data interoperability considering different data formats (Challenge 1) and a good number of custom requirements that can be elicited for enabling data interoperability across heterogenous systems (Challenge 2). In the following sections, we discuss how Chimera can help in addressing the mentioned challenges and how it can be used to define a semantic data transformation pipeline for the TANGENT scenario.

## 3. Chimera

The Chimera framework aims at addressing the two mentioned challenges by (i) providing an efficient set of software components to implement an any-to-one centralized mapping approach through Semantic Web technologies, and (ii) integrating these components in a broader ecosystem facilitating the implementation and deployment of semantic converters among heterogeneous systems. The design of the Chimera framework follows a modular and low-code approach to minimise the effort required to specify and configure a semantic data transformation pipeline for different scenarios.

The decision of the modular approach is based on the assumption that a semantic data transformation pipeline could be broken down into a set of smaller, composable and reusable building blocks. Moreover, the aim is to facilitate the extensibility of the framework to integrate additional blocks. We defined four types of building blocks for a semantic data transformation pipeline (shown in Figure 2):

1. **Graph Construction**. Data from heterogeneous sources is converted to RDF according to a reference ontology. The conversion process from a specific data format to RDF, also known as lifting, can be handled by employing different approaches for knowledge graph construction [3], such as RML[6] mappings execution.
2. **Graph Transformation**. An RDF knowledge graph can be augmented by means of graph operations such as: adding RDF triples to the graph, generating a new graph through SPARQL Construct queries, and applying inference considering the reference ontology.
3. **Graph Validation**. The validation of the graph (e.g., by using SHACL shapes) can be useful and necessary to validate the correctness of the implemented procedure and/or of the ingested data.
4. **Graph Exploitation**. The information in an RDF knowledge graph is extracted and converted to a specific target data format through a lowering process.

Despite being presented in order, each operation is decoupled and defined to support different pipelines. For example, the result of the pipeline can be an RDF graph constructed and then transformed, thus omitting a *Graph Lowering* operation.

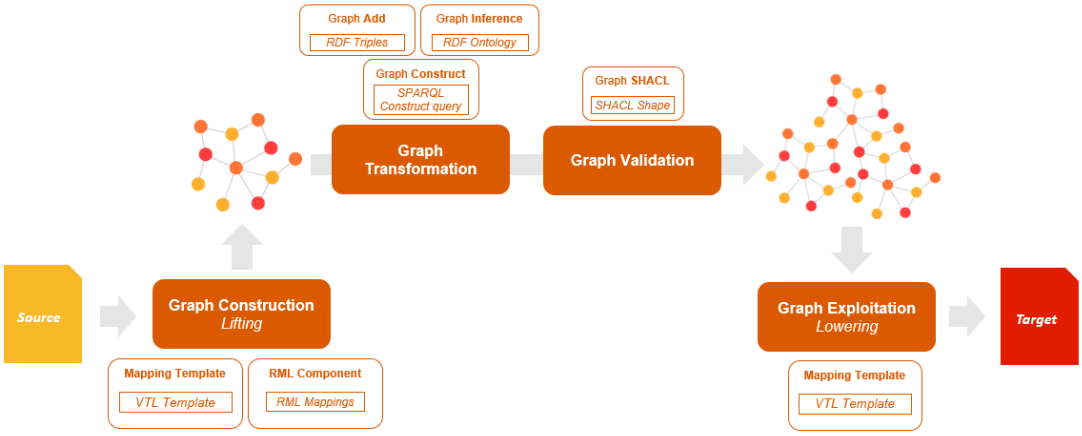


**Figure 2:** Overview of the steps of a semantic data transformation pipeline and available Chimera components.

We implemented Chimera on top of Apache Camel[4], a Java integration framework to facilitate the integration with various systems consuming or producing data. We chose Camel to inherit its features and advantages: not only it is a completely open-source, configurable and extensible solution, but it also implements best practices and patterns to solve the most common integration problems, including the Enterprise Integration Patterns [7]. Finally, being a robust and stable project, Camel supports out-of-the-box several components, runtimes and formats to access and integrate a large set of existing systems and environments.

Apache Camel relies on the basic concept of *Route* defining a certain logic to load, extract, integrate, transform and output data. Each *Route* is a pipeline composed of a set of components that are applied in a specific sequence to a certain *Exchange*, i.e., an entity going through a *Route*. The *Exchange* is identified by an identifier and it is similar to an envelope, it contains the messages (e.g., the data being processed) but also a set of properties that can be used to carry an additional state during the *Route* execution.

The basic idea of Chimera is to define additional components[5] for Apache Camel supporting operations on an RDF graph that is passed along the Camel *Route*. All Chimera components use the RDF4J library[6] to process and handle RDF graphs. The *RDF Graph* in Chimera pipelines is an abstraction that can refer to a local knowledge graph (in-memory, filesystem), or a remote graph stored in a triplestore or accessible through a SPARQL endpoint. Chimera components can be used also in conjunction with already available and established components from the Apache Camel framework[7] and/or custom components defined by the user (e.g., for a custom pre-processing). Moreover, the definition of additional components for each graph operation is possible (e.g., the integration of an additional solution for knowledge graph construction) by simply leveraging the *RDF Graph* abstraction.

[4]https://camel.apache.org/
[5]https://camel.apache.org/manual/writing-components.html
[6]https://rdf4j.org/
[7]https://camel.apache.org/components/3.20.x/index.html

Chimera defines the following set of components to enable the graph operations described:

**Graph Component:** The graph component allows for the following operations:

- **Graph Get:** accesses the RDF Graph to be used in the pipeline.
- **Graph Add:** adds RDF triples from one or more resources to the RDF Graph.
- **Graph Construct:** applies a SPARQL Construct query to the RDF Graph giving to the user the possibility of adding the generated triples to the previous RDF Graph or to a new RDF Graph.
- **Graph Inference:** enables inference on the RDF Graph considering a given ontology (currently RDFS [8] is supported).
- **Graph SHACL:** validates the RDF Graph using a (set of) SHACL shape(s).
- **Graph Detach:** clears the RDF Graph and/or closes the pending connections.
- **Graph Dump:** writes the RDF Graph to a file in a user-specified format, e.g. Turtle, N3 and others.

All the operations implemented by the graph component support the execution considering specific named graphs.

**RML Component:** a lifting block enabling the execution of RML [6] mappings for knowledge graph construction from heterogeneous data sources. The component integrates the rmlmapper-cefriel[8], a fork of the rmlmapper library[9] [5].

**Mapping Template Component:** both lifting and lowering block, it is based on Apache Velocity[10] to implement a template-based solution to query the input data, process the result set and generate the output in the desired format[11]

The current version of Chimera is the result of a complete refactoring effort over the initially designed framework[1] and integrates the feedback collected from several EU research (e.g., SPRINT[12], RIDE2RAIL[13], TANGENT) and innovation projects. In particular, this version facilitates the reuse of the components and the extensibility of the framework by improving their decoupling and harmonising their configuration parameters. As an example, the *Chimera Resources* abstraction has been introduced to generalise the specification of resources that Chimera components may need for their configuration. Examples of such resources are the mapping files that define the graph construction process. Each resource may be locally available (classpath or filesystem) or retrieved from a remote repository through the HTTP(S) protocol.

[8]This implementation is also referred to as Chimera in the literature about RML processors [3, 9], but represents one of the building blocks of the overall Chimera framework. Link to the repository: https://github.com/cefriel/rmlmapper-cefriel

[9]https://github.com/RMLio/rmlmapper-java

[10]https://velocity.apache.org/

[11]Integrates the https://github.com/cefriel/rdf-template library that can also be used as a standalone tool

[12]http://sprint-transport.eu/

[13]https://ride2rail.eu/

```yaml
- beans:
    - name: someTriples
      type: com.cefriel.util.ChimeraResourceBean
      properties:
        url: "https://some_address"
        sertalizationFormat: "turtle"
    - name: mappingFile
      type: com.cefriel.util.ChimeraResourceBean
      properties:
        url: "file://absolute_path_to_mappingFile"
- from:
    uri: "graph://get?resources=#bean:someTriples"
    steps:
     - to:
        uri: "rml://?mapping=#bean:mappingFile"
     - to:
        uri: "seda:writeToDatasink"
```

**Figure 3:** Example of YAML configuration for a Chimera pipeline.

A Camel *Route* can be defined by simply providing a configuration file through one of the available DSL languages[14], without the need of writing additional code. Figure 3 shows an example Chimera pipeline configured using the YAML DSL of Apache Camel. In the example route: (i) an RDF resource is accessed via HTTP and integrated into an RDF Graph, (ii) a lifting operation is executed on the incoming message considering an RML declarative mapping file from the local filesystem and the result is added to the RDF Graph, (iii) the result is sent to a separate route (e.g., writing the output to a specific data sink). Thanks to the huge set of Camel's predefined components, Chimera can enable both *batch conversion* and *service mediation* scenarios by leveraging multiple input and output channels. For example, input data from this route can be acquired by polling directories or FTP servers, or by exposing REST services, Web APIs and SOAP services, by using different publish-subscribe technologies such as JMS or AMQP.

## 4. Chimera in action

The Chimera framework is available on GitHub[15] and is released under the Apache 2.0 license. The repository contains the three components discussed and can be integrated into any Apache Camel project.

A tutorial showing an example project to construct semantic data transformation pipelines is also available on GitHub[16]. The tutorial aims at introducing each Chimera component and showing how to configure them through the Apache Camel Spring XML DSL. For demonstrative purposes, a sample GTFS[17] feed is considered as input and the Linked-GTFS ontology[18] is used as the reference ontology. Multiple pipelines that make use of the Chimera components are presented and described in detail. Each pipeline is exposed as an HTTP endpoint, showing

[14]https://camel.apache.org/manual/dsl.html
[15]https://github.com/cefriel/chimera
[16]https://github.com/cefriel/chimera-tutorial
[17]https://developers.google.com/transit/gtfs
[18]https://github.com/OpenTransport/linked-gtfs

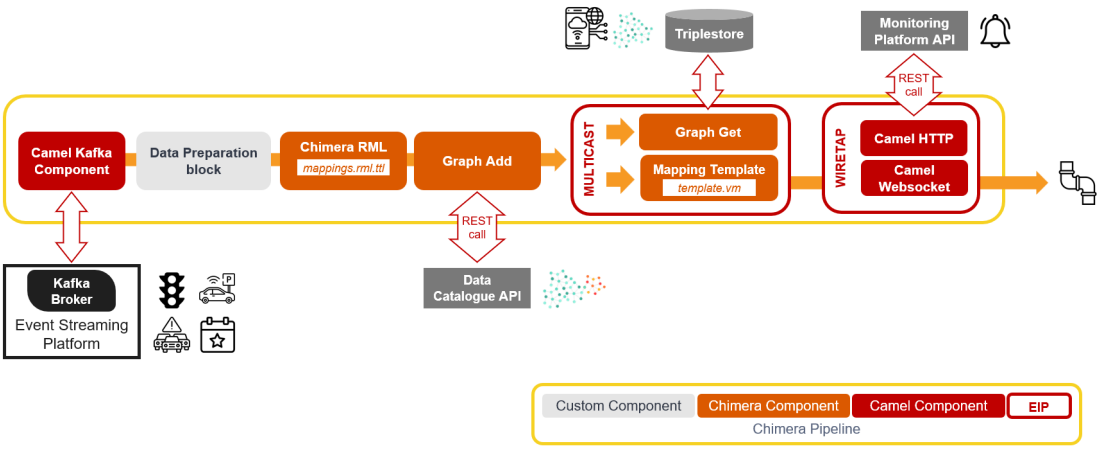


**Figure 4:** Chimera pipeline for the TANGENT use case presented.

also the integration with the HTTP Apache Camel components. The tutorial also exemplifies different deployment options for Chimera pipelines, such as standalone JAR and/or container-based execution. Notably, in the proposed approach the defined Chimera pipelines can be modified through the XML file without requiring a new build process.

To exemplify the potentiality of the Chimera approach for data interoperability across heterogeneous systems, Figure 4 describes the configuration of a pipeline fulfilling the requirements for the TANGENT use case presented in Section 2. The Apache Camel Kafka component[19] is able to ingest data from a Kafka broker and can be directly incorporated into a Chimera pipeline. The data coming from Kafka may require an additional data preparation step, in this case, the user can integrate into the pipeline a custom Apache Camel component defined for the specific use case. The benefit of defining such logic as a Camel Component is that once it has been defined, then it can be reused in other pipelines. For the harmonisation of incoming data using a common ontological data model, the Chimera RML component is integrated into the pipeline and configured to execute RML mappings for knowledge graph construction. The harmonised RDF data can then be augmented with RDF triples from external resources, such as a Data Catalogue, by means of the *Graph Add* operation. Using the *multicast* Enterprise Integration Pattern (EIP) provided by Apache Camel, the result of previous processing can be forwarded in parallel to multiple destinations. For example, it can be sent to a remote triplestore for storing but also continue the processing into the pipeline. Through the Mapping Template component, the knowledge graph is lowered to the target JSON format. The data in the obtained format can then be forwarded to the relevant components using the Camel-provided Wiretap EIP that allows to: (i) asynchronously process and send data to the monitoring platform, (ii) synchronously forward the data to their final destination using the Camel WebSocket component[20].

[19] https://camel.apache.org/components/3.20.x/kafka-component.html
[20] https://camel.apache.org/components/3.20.x/websocket-component.html

## 5. Related Works

The survey from Van Assche et al. [3] defines two types of transformation, i.e., data transformations and schema transformations. Chimera aims at covering both types of transformation with the objective of enabling data interoperability among stakeholders through an any-to-one semantic conversion mechanism. Whilst only two approaches for the *lifting* are currently implemented in Chimera, the approach can be extended by considering each of the tools mentioned in the survey that is able to generate an RDF Graph. Once defined the graph, all the other Chimera components can be leveraged to define a pipeline.

Considering the lowering, a limited set of standardised solutions exists in the literature. Potential approaches were proposed in [10], by querying RML lifting mappings to recreate a target CSV format, and in [11], by relying on SPARQL to define RDF transformation rules. In our work, we adopt a declarative approach to implement lowering, relying on SPARQL for querying data and on a template engine to efficiently serialise data according to the target format. Any approach that allows specifying a transformation from an RDF graph to a target format can be integrated as a Chimera component.

Finally, we cite similar works in the literature defining semantic-based ETL ("Extract, Transform and Load") pipelines. Talend4SW[21], similarly to the approach of Chimera with Apache Camel, leverages an existing tool (Talend) and offers additional components to interact with RDF data. UnifiedViews [12], LinkedPipes [13] and Barnard59[22] exploit Semantic Web principles to feed and curate RDF knowledge bases, thus focusing on the graph construction, transformation and validation. Each of mentioned approaches has its advantages for specific use cases, Chimera focuses on the definition of pipelines for data integration and interoperability among heterogeneous systems.

## 6. Conclusions and future work

In this paper, we presented the Chimera framework that addresses the problem of data interoperability in a multi-stakeholder ecosystem through the definition of semantic data transformation pipelines relying on a reference ontology. The proposed approach exploits an any-to-one centralised mechanism to limit the number of mappings to be defined and has the advantage of generating an integrated and interoperable knowledge graph. The Chimera framework has been implemented by extending Apache Camel to offer a flexible and configurable solution for integrating semantic data transformation pipelines across heterogeneous systems. The framework reduces the effort required to specify custom pipelines and can take advantage of the overall production-ready ecosystem of Camel components. Chimera specifically defines Camel components addressing four operations on RDF graphs: graph construction, graph transformation, graph validation and graph exploitation. Finally, the usage of Chimera was exemplified through a transportation domain conversion use case from the TANGENT project.

In future work, we would like to explore the integration of Chimera into existing graphical

[21] Talend4SW, cf. https://github.com/fbelleau/talend4sw.
[22] https://github.com/zazuko/barnard59

tools for the construction and visualization of Apache Camel pipelines, e.g., Apache Karavan[23]. Such integration would simplify the definition of semantic data transformation pipelines also by suggesting the set of available parameters and configurations for each component. Moreover, within the SmartEdge[24] project we will investigate the adoption of the presented approach on the edge (resource-constrained devices) and on the cloud (to offer high scalability), e.g., by investigating the ecosystem of deployment runtimes supported by Apache Camel and/or by defining new Chimera components.

## Acknowledgments

The presented research was partially supported by the TANGENT project (Grant Agreement 881825), funded by the European Commission under the Horizon 2020 Research and Innovation Programme, and the SmartEdge project, funded under the Horizon Europe RIA Research and Innovation Programme (Grant Agreement 101092908).

## References

[1] M. Scrocca, M. Comerio, A. Carenini, I. Celino, Turning transport data to comply with EU standards while enabling a multimodal transport knowledge graph, in: Proceedings of the 19th International Semantic Web Conference, volume 12507, Springer, 2020, pp. 411–429. doi:10.1007/978-3-030-62466-8_26.

[2] G. Vetere, M. Lenzerini, Models for semantic interoperability in service-oriented architectures, IBM Systems Journal 44 (2005) 887–903. doi:10.1147/sj.444.0887.

[3] D. Van Assche, T. Delva, G. Haesendonck, P. Heyvaert, B. De Meester, A. Dimou, Declarative rdf graph generation from heterogeneous (semi-)structured data: A systematic literature review, Journal of Web Semantics (2022) 100753.

[4] M. Sadeghi, P. Buchníček, A. Carenini, O. Corcho, S. Gogos, M. Rossi, R. Santoro, et al., SPRINT: Semantics for PerfoRmant and scalable INteroperability of multimodal Transport, in: 8th Transport Research Arena TRA 2020, 2020, pp. 1–10. URL: http://hdl.handle.net/11311/1132635.

[5] M. Scrocca, A. Carenini, M. Comerio, I. Celino, Semantic Conversion of Transport Data Adopting Declarative Mappings: An Evaluation of Performance and Scalability, in: D. Chaves-Fraga, P. Colpaert, M. Sadeghi, M. Scrocca, M. Comerio (Eds.), Proceedings of the 3rd International Workshop Semantics And The Web For Transport, volume 2939 of *CEUR Workshop Proceedings*, CEUR, Online, September, 2021. URL: https://ceur-ws.org/Vol-2939/#paper2, iSSN: 1613-0073.

[6] A. Dimou, M. V. Sande, P. Colpaert, R. Verborgh, E. Mannens, R. V. de Walle, RML: A generic language for integrated RDF mappings of heterogeneous data, in: Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), volume 1184, CEUR-WS.org, 2014. URL: http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[23] https://github.com/apache/camel-karavan
[24] https://www.smart-edge.eu/

[7] G. Hohpe, B. Woolf, Enterprise integration patterns: Designing, building, and deploying messaging solutions, Addison-Wesley Professional, 2004.

[8] D. Brickley, R. V. Guha, B. McBride, Rdf schema 1.1, W3C recommendation 25 (2014) 2004–2014.

[9] J. Arenas-Guerrero, M. Scrocca, A. Iglesias-Molina, J. Toledo, L. Pozo-Gilo, D. Doña, Ó. Corcho, D. Chaves-Fraga, Knowledge graph construction with R2RML and RML: an ETL system-based overview, in: D. Chaves-Fraga, A. Dimou, P. Heyvaert, F. Priyatna, J. F. Sequeda (Eds.), Proceedings of the 2nd International Workshop on Knowledge Graph Construction co-located with 18th Extended Semantic Web Conference (ESWC 2021), Online, June 6, 2021, volume 2873 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021. URL: http://ceur-ws.org/Vol-2873/paper11.pdf.

[10] C. Allocca, A. Gougousis, A preliminary investigation of reversing rml: From an rdf dataset to its column-based data source, Biodiversity data journal 3 (2015) e5464. doi:10.3897/BDJ.3.e5464.

[11] O. Corby, C. F. Zucker, F. Gandon, SPARQL template: a transformation language for RDF, Ph.D. thesis, Inria, 2014.

[12] T. Knap, M. Kukhar, B. Macháč, P. Škoda, J. Tomeš, J. Vojt, Unifiedviews: An etl framework for sustainable rdf data processing, in: European Semantic Web Conference, Springer, 2014, pp. 379–383.

[13] J. Klímek, P. Škoda, M. Nečaskỳ, Linkedpipes etl: Evolved linked data preparation, in: European Semantic Web Conference, Springer, 2016, pp. 95–100.