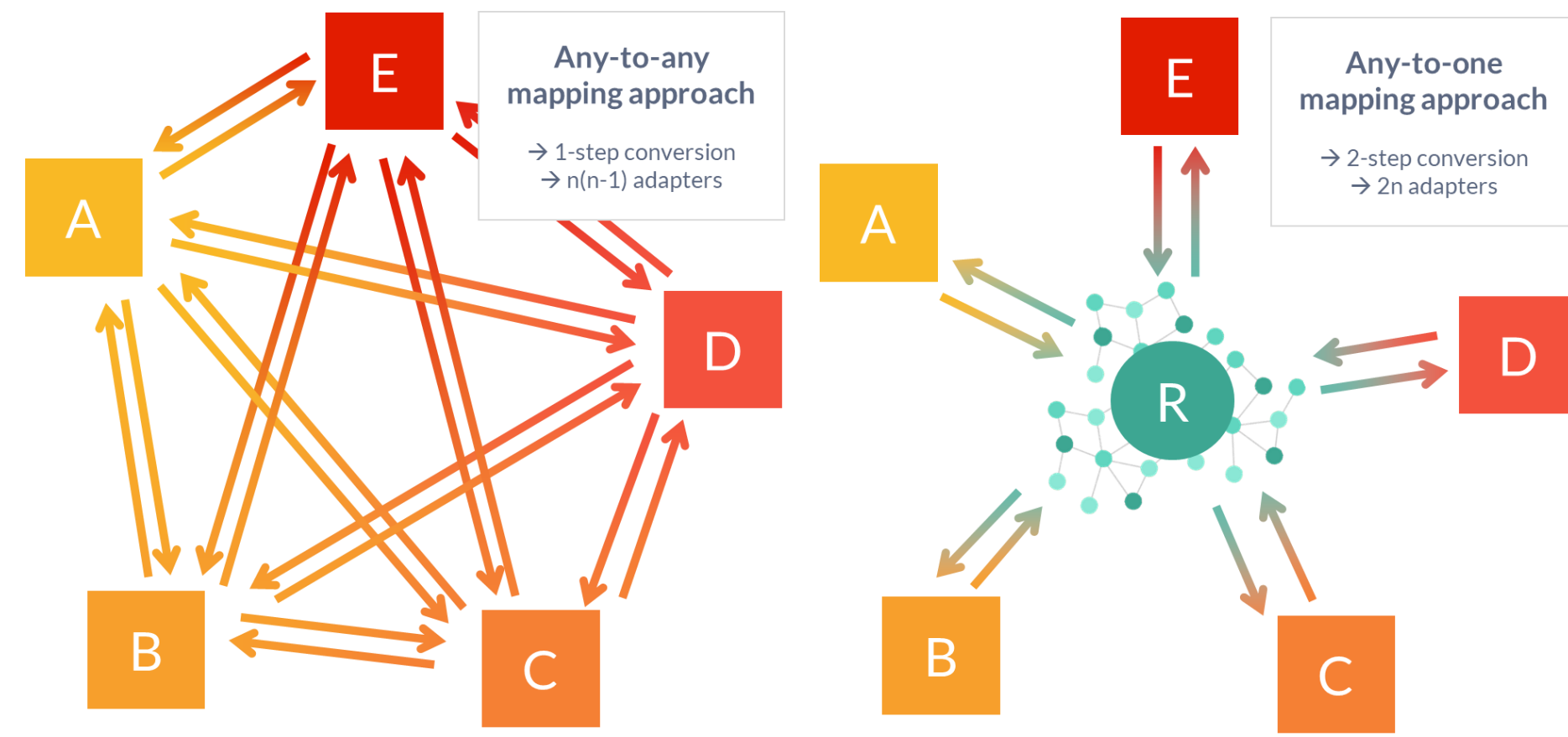


Can Knowledge Graphs support data interoperability in a multi-stakeholder ecosystem?

Data exchange does not define interoperability. Two or more systems are **interoperable** if they **can communicate** and exchange information in an effective way and **without loss of meaning**.

Multi-stakeholder ecosystem:

- Many **different data formats**
- **Heterogeneous** specifications and **semantics**
- **Point-to-point** system integration **doesn't scale**

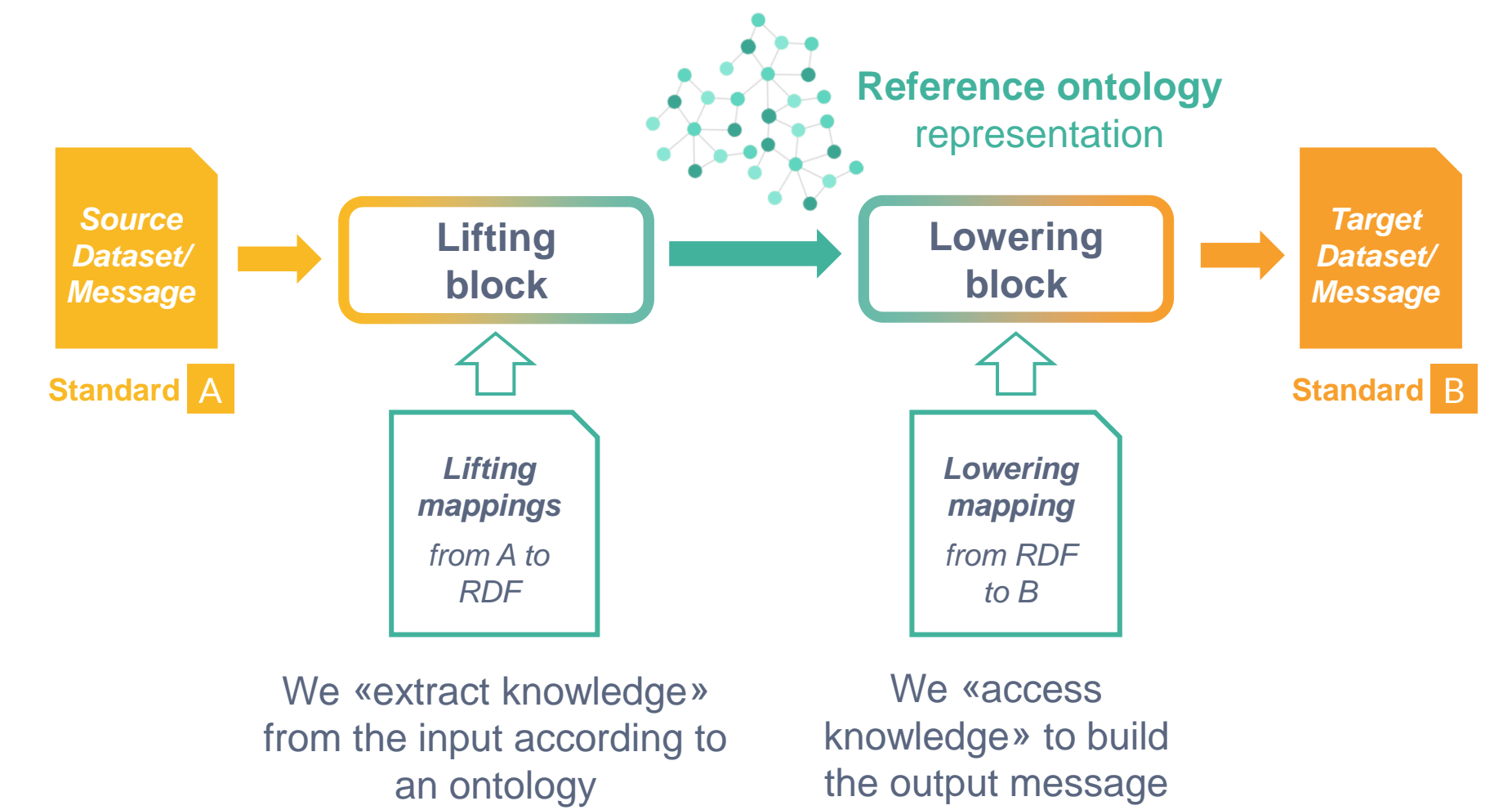


Any-to-one mapping through Knowledge Graphs

Declarative mapping rules can enable the conversion between different formats/standards sharing common semantics¹

- ✓ Keep current legacy systems
- ✓ Only need to define mappings to/from a *reference ontology*
- ✓ *Interoperable and integrated Knowledge Graph* as an additional valuable product of the conversion

Semantic Data Conversion



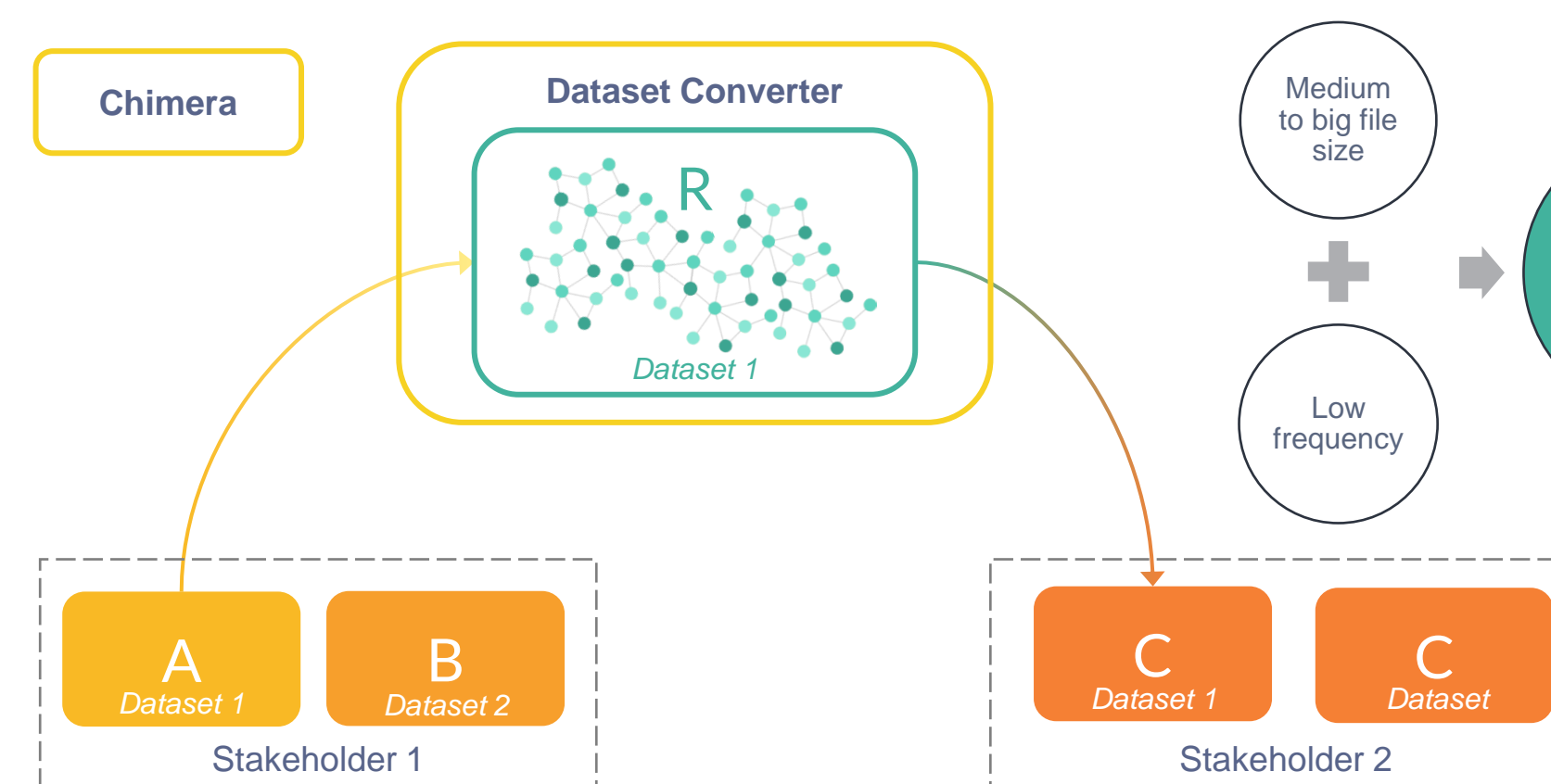
What are the requirements to support data interoperability across heterogeneous information systems?

There is no single *interoperability problem* and, therefore, no single *interoperability solution*. A flexible and configurable set of specialized tools is needed to cope with **different functional requirements for the integration of semantic converters across heterogeneous information systems**.

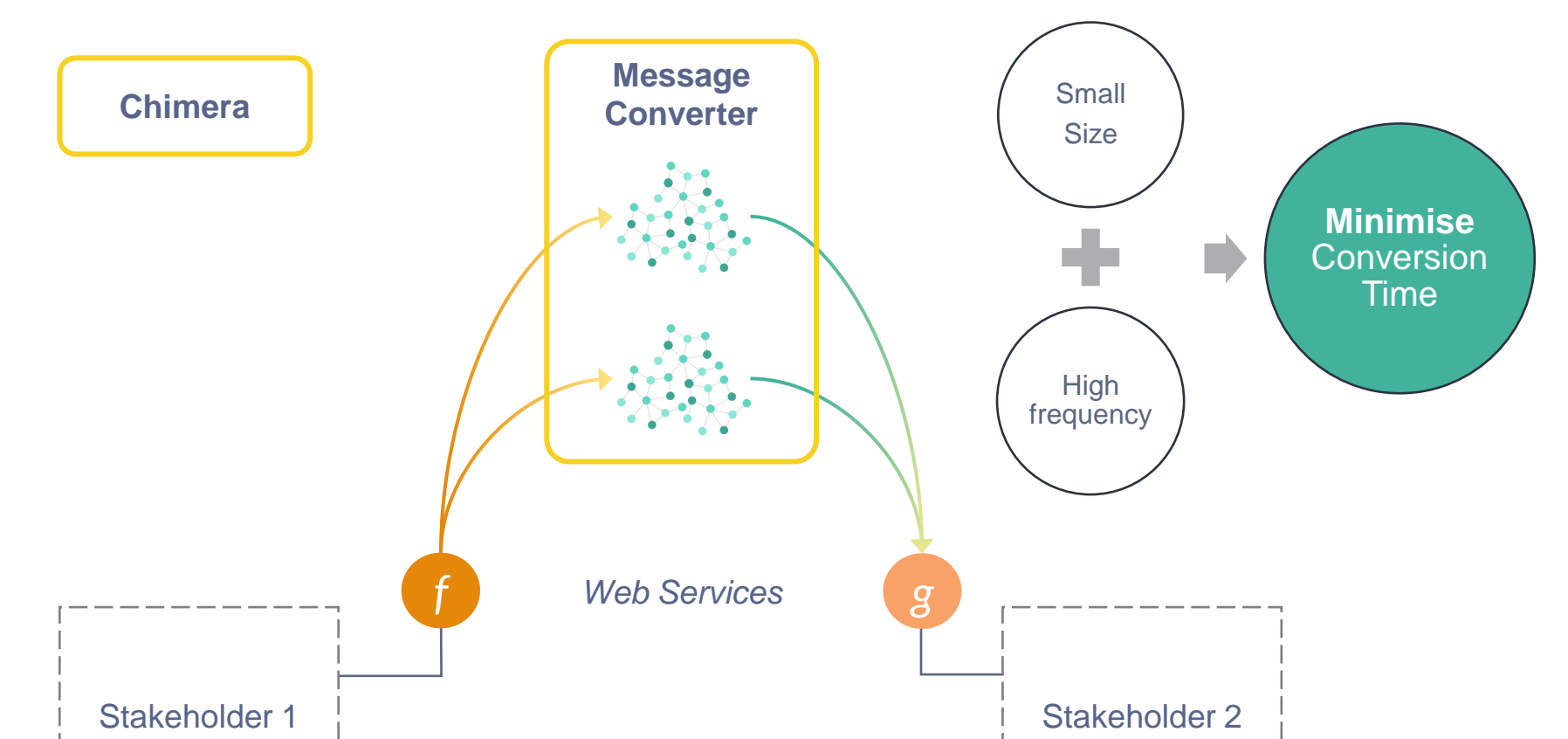
Two main use cases with different **non-functional requirements** can be identified:

- Interoperability of **datasets** in different formats
- Interoperability of **web services** relying on different API specifications

Batch Conversion Use Case

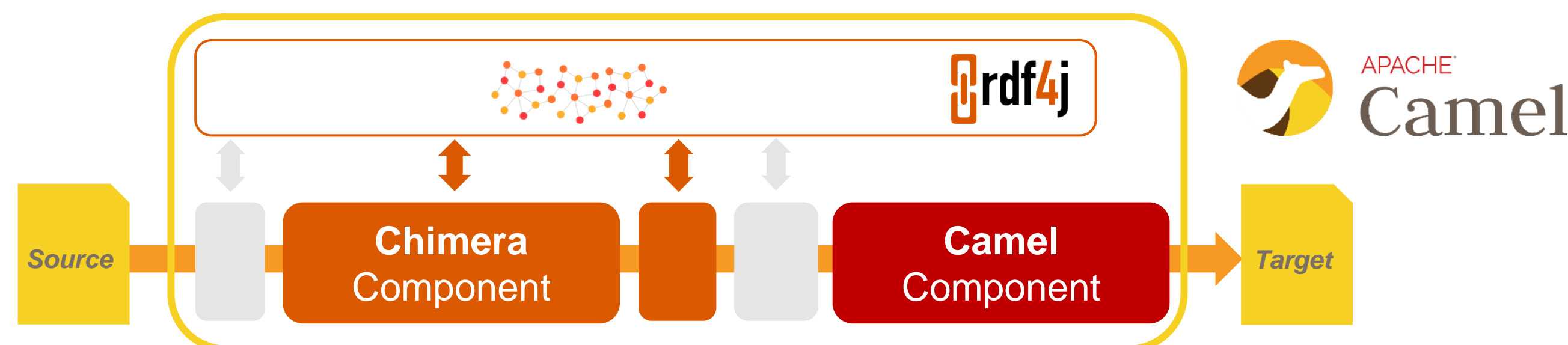


Service Mediation Use Case



Chimera: a modular and configurable solution for semantic data transformation pipelines

Chimera is a **modular** and **configurable** solution for the Apache Camel integration framework to minimise the effort required to specify and configure a **semantic data transformation pipeline**



Chimera RML

RMLMapper fork⁴



Knowledge Graph construction through **RML** mapping rules

Chimera Lowerer

rdf-template⁵



Knowledge extraction using Apache Velocity **templates** with embedded **SPARQL** queries

Chimera Component

Chimera components implement (i) *operations on RDF Graphs*, (ii) *Lifting*, and (iii) *Lowering*

Camel Component

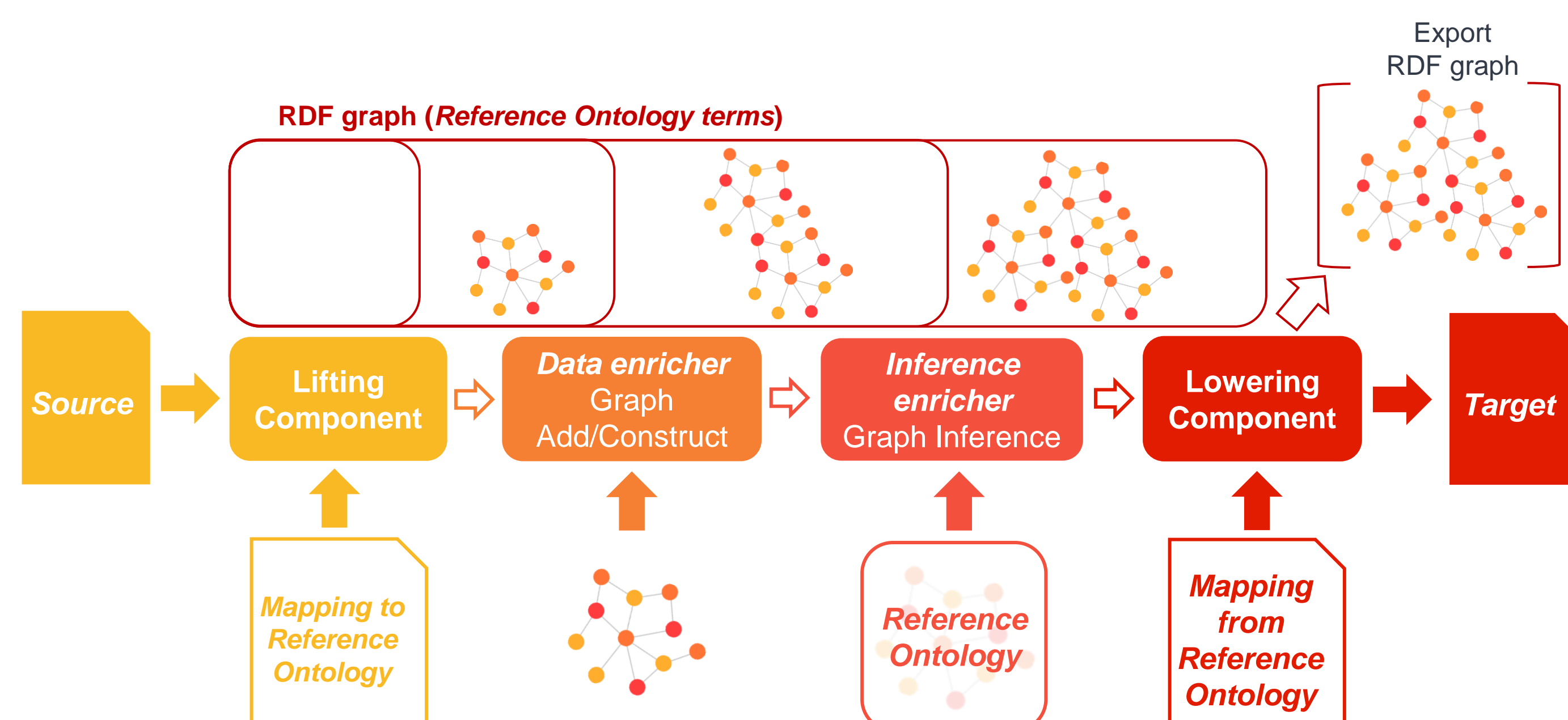
Production-ready **Camel components** can be used to implement *Enterprise Integration Patterns* and *data source/sink integration*

EIP

Custom Component

Integration with existing ETL pipelines and custom components (e.g., for pre-processing, validation) can be easily performed

A generic **semantic data transformation pipeline** enabled by Chimera, can **combine different components and operations** to fulfil **heterogenous requirements**. The usage of a remote triplestore is supported.



The minimum Chimera pipeline to implement a **Semantic Data Converter** is composed of a **Lifting** and **Lowering Component**.

Chimera RML

This approach exploits mappings defined through the **RDF Mapping Language (RML)**, i.e., mapping rules from different data formats to an RDF serialization

Chimera RML supports mappings from **heterogeneous data sources** (CSV, XML, JSON)

Built-in support: (i) data enrichment between **multiple input data sources** (also with different data formats); (ii) apply **custom functions** in processing the input data

Optimizations implemented: stream support, input parsers, concurrent processing, incremental writing of materialised triples

Compared with **state-of-the-art** knowledge graph materializers³

```
<AuthorityMapping>
a rr:TriplesMap;

rr:logicalSource [
  rr:source "agency.csv";
  rr:referenceFormulation ql:CSV;
];

rr:subjectMap [
  rr:template "agencies/{agency_id}";
  rr:class tm:Authority;
];

rr:predicateObjectMap [
  rr:predicate tm:name;
  rr:objectMap [
    rr:reference "agency_name"
  ]
];

rr:predicateObjectMap [
  rr:predicate tm:id;
  rr:objectMap [
    rr:reference "agency_id"
  ]
].
```

Chimera Lowerer

A declarative approach based on **templates** to guarantee maximum flexibility on the output format

It exploits **Apache Velocity** templates (<https://velocity.apache.org>) replacing at runtime variables with actual values

SPARQL queries allows defining in the template how to **access an RDF Graph**

Velocity Template Language (VTL) allows defining in the template how to manipulate results obtained from queries and fill the template to **generate the expected output data format**

```
#set ( $query = "
SELECT ?id ?name
WHERE {
  ?a <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <https://w3id.org/transmodel/terms#Authority> .
  ?a <https://w3id.org/transmodel/terms#name> ?name .
  ?a <https://w3id.org/transmodel/terms#id> ?id . } )

#set ( $authorities = $reader.executeQuery($query) )
<?xml version="1.0" encoding="iso-8859-1"?>
<PublicationDelivery version="1.0"
  xsi:schemaLocation="http://www.netex.org.uk/netex/../../xsd/
  xmlns="http://www.netex.org.uk/netex" xmlns:xsi="http://www.w
  <dataObjects>
    <ResourceFrame>
      <organisations>
        #foreach($authority in $authorities)
          <Authority id="$authority.id">
            <Name>$authority.name</Name>
          </Authority>
        #end
      </organisations>
    </ResourceFrame>
  </dataObjects>
</PublicationDelivery>
```

Performance and scalability evaluation

Configuration: Docker Container on CentOS Linux 7, 8-core CPU with 24 GB Memory limit.

Batch Conversion

Datasets and Mappings: GTFS-Madrid Benchmark datasets (formats CSV, JSON, XML) and lifting/lowering mappings to/from Linked GTFS from/to GTFS.

Performance: Conversion time for Scale 1 (5MB input, 500K triples generated)

Scalability: tested w.r.t. the size of the input dataset: scale 1,5,10,50,100

Results²

Performance: **CSV 10.83s, JSON 30.89s, XML 16.26s**. Improvement w.r.t. *RMLMapper v4.7*: CSV (2x), JSON (1.6x) and XML (>1000x).

Scalability: Able to convert CSV, JSON and XML datasets up to **100MB generating 18M triples**.

Service Mediation

Datasets and Mappings: 50KB input message. Custom mappings from input format to reference ontology, and from reference ontology to output format.

Performance: Conversion time for a single request

Scalability: tested w.r.t. number of concurrent requests (*scale*: 10, 50, 100, 150, 500, 1000, 2500, 5000; *ramp-up period*: 1 second; *loop count*: 1)

Results²

Performance: **140ms for a single request**. Improvement w.r.t naive implementation: 5x.

Scalability: **100 requests/s** for a single instance of the converter

References

¹ Scrocca M., Comerio M., Carenini A., Celino I. (2020) **Turning Transport Data to Comply with EU Standards While Enabling a Multimodal Transport Knowledge Graph**. In: The Semantic Web – ISWC 2020. Springer. https://doi.org/10.1007/978-3-030-62466-8_26

² Scrocca M., Carenini A., et al. (2021) **Semantic Conversion of Transport Data Adopting Declarative Mappings: An Evaluation of Performance and Scalability**. In: Sem4Tra 2021 @SEMANTICS. CEUR-WS.org. <http://ceur-ws.org/Vol-2939/paper2.pdf>

³ Arenas-Guerrero, J., Scrocca, M et al. (2021) **Knowledge graph construction with R2RML and RML: An ETL system-based overview**. In: KGC 2021 @ESWC. CEUR-WS.org. <http://ceur-ws.org/Vol-2873/paper11.pdf>

⁴ Fork based on the *rml-mapper* <https://github.com/cefriel/rmlmapper-cefriel>

⁵ *rdf-template* library <https://github.com/cefriel/rdf-template>