Contract No. H2020 – 826172

SEMANTICS FOR PERFORMANT AND SCALABLE INTEROPERABILITY OF MULTIMODAL TRANSPORT

D5.3 – Validation of pilot implementation (C-REL)

Due date of deliverable: 30/02/2020

Actual submission date: 30/10/2020

Leader/Responsible of this Deliverable: Cefriel

Reviewed: Y

Document status		
Revision	**Date**	**Description**
0.1	08/01/2020	Draft, working version
0.2	06/05/202	Added performance and scalability results
0.3	08/05/2020	Completed the section about Functional Validation
0.4	11/05/2020	Added Executive summary and requirements for F-Rel after the analysis of the functional evaluation
1.0	25/05/2020	Complete version, ready for TMC approval
1.1	23/10/2020	Revised version ready for resubmission
1.2	30/10/2020	Final version after TMC approval

Project funded from the European Union's Horizon 2020 research and innovation programme		
Dissemination Level		
PU	Public	X
CO	Confidential, restricted under conditions set out in Model Grant Agreement	
CI	Classified, information as referred to in Commission Decision 2001/844/EC	

Start date of project: 01/12/2018 Duration: 25 months

EXECUTIVE SUMMARY

The aim of the deliverable D5.3 is to validate the functional and nonfuctional capabilities of the implemented solution within the initial release (C-REL). The part of this deliverable is also the assessment of the realized architecture with respect to quantitative and qualitative indicators, including the KPIs defined within WP3 Performance and Scalability of the Interoperability Framework. In particular, the document contains the outcomes of the functional and the performance and scalability evaluation according to the scenarios defined in Task 5.1 “Design of proof-of-concept scenarios”. Because of the process of the design-implementation-validation of the proof-of-concept consists two steps (two releases), the requirements to be addressed for the final version (the second release) are defined in this deliverable.

To ease understanding the link between requirements and how they have been validated, Annex A contains a summary of all requirements (high level IF requirements, IF components functional requirements, IF components performance and scalability requirements).

ABBREVIATIONS AND ACRONYMS

Abbreviation	Description
C-REL	Core Release
EU	European Union
GA	Grant Agreement
H2020	Horizon 2020 framework programme

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

This deliverable provides an evaluation of the first proof of concept of the Interoperability Framework as designed in D3.3 “Design of Architecture, Testing Infrastructure, Test Cases and Benchmarks of the IF (C-REL)”. The document is divided into three main sections. The initial section covers the functional evaluation of the scenarios defined in D5.1 “Requirements, scenarios and use cases for the proof-of-concept (C-REL)”. In that section, we have described how to configure the SPRINT tools and how to use them in order to implement each scenario.

The second section contains the performance and scalability evaluation of the SPRINT tools. As anticipated in D3.3, the evaluation of the tools focused on aspects related to “Interoperability execution”. Among all the tools which are being developed in SPRINT, some of them are “support” tools that can be used to improve the quality of the mappings (Mapping suggester) or to help govern the IF Ecosystem, while others (like Converters and Resolvers) are meant as services to be invoked frequently to achieve interoperability. Since performances and scalability of the latter category have the greatest impact on achieving an effective interoperability, we decided to focus the analyses described in this document on the SPRINT solutions to help create Converters and Resolvers.

Finally, the last section provides an analysis of the outcomes of the evaluation, providing requirements to be addressed in the final version of the proof of concept.

2. FUNCTIONAL VALIDATION

The scenarios of this section cover various use cases for the basic functions of the IF, which involve the following components/tools of the IF:

Table 1: List of Involved Component/tools of the IF for basic function scenarios

IF Component/Tools	Use Case Scenario
Asset Manager	Scenario S3 : Service/Asset Discovery (Simple Discovery) Scenario S4: Service/Asset Discovery (Distributed SPARQL endpoints) Scenario S5: Direct Access use case for Batch Data Conversion Scenario S6: 1.1 Direct download for runtime data/message conversion Scenario S8: Automatic converter building Scenario S9: Fast Adaptation to Peaks
User Manager	Scenario S1 : Joining the IF Use case (User Registration) Scenario S2 : Joining the IF Use case (Provider Registration)
SPARQL endpoint	Scenario S4: Service/Asset Discovery (Distributed SPARQL endpoints)
Asset Discovery	Scenario S3: Service/Asset Discovery (Simple Discovery)

For each scenario, we will explain how the various tools have been used to validate the fulfillment of the goals.

2.1 SCENARIO S1: JOINING THE IF USE CASE (PROVIDER REGISTRATION)

Actor	**HT-train (TSP):** a train service provider
Target Component/Sub-system/Entity	User Management
Description	This scenario illustrates the registration process for a user that intends to join IF as a “Service Provider”
Story	Bill is an employee in HT-train which is responsible to register this operator to IF. He goes to IF website of Italy *[IF].it* and selects to register to IF as Service Provider Role. To create an account, he inserts all the required information related to himself as well as his company, including username, password, type of the company, etc. After successful registration and confirmation of the identity of the registered user, he is then redirected to the **Back-Office** view of **Asset Manager (AM). Back-Office** is conceived as the provider's panel in IF and presents required interfaces to the functionalities available for a service provider such as Asset Registration.

2.1.1 Tools configuration

To validate this scenario, we created the Administrator user. This operation must be performed outside of the Web applications, and can be achieved by executing the command

```
docker-compose -f production.yml run --rm django python manage.py createsuperuser
```

Once the Administrator user has been created, we created two user Groups using the Django Web interface, namely “Provider” and “Consumer”.

2.1.2 Validation

To verify the possibility to execute this scenario, we simulated the registration process of the Provider user “Bill”.

Bill wants to contribute to the Shift2Rail ecosystem, therefore he tries to open the Publisher Web application. Since he's neither logged in nor registered, he is redirected to the login/registration page. Bill decides to register himself and fills in the registration form, as depicted in Figure 1.

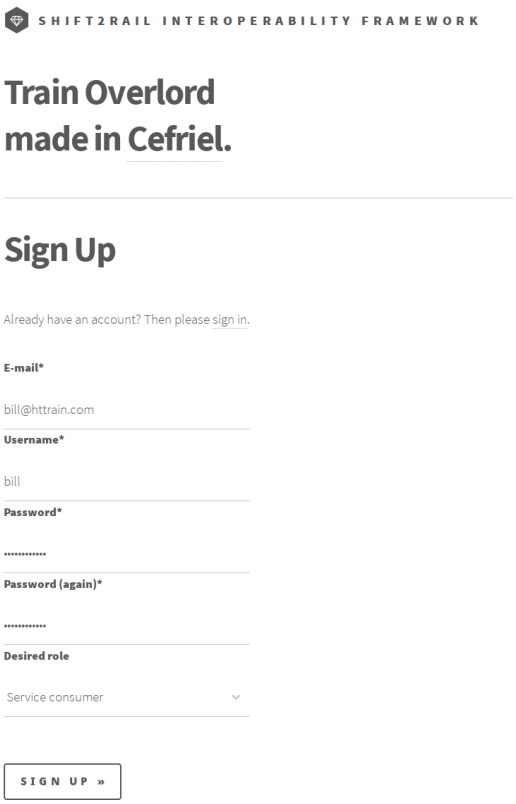


Figure 1: Service consumer registration form

Once Bill clicks on the "Register" button, the Administrator user is notified via email (as shown in) about the registration attempt.

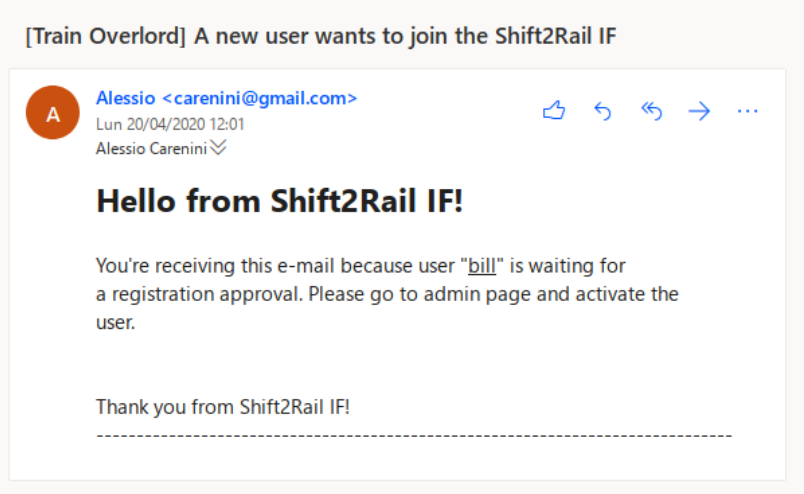


Figure 2: Email notifying the registration request

The Administrator user then opens the Asset Manager administration page and opens the “Users” page. The Administrator user then assigns the role “Provider” to Bill, and “activates” him allowing logins, as illustrated in Figure 3. Bill is then notified about the activation of his credentials, and he will be able to log into the Publisher web application.

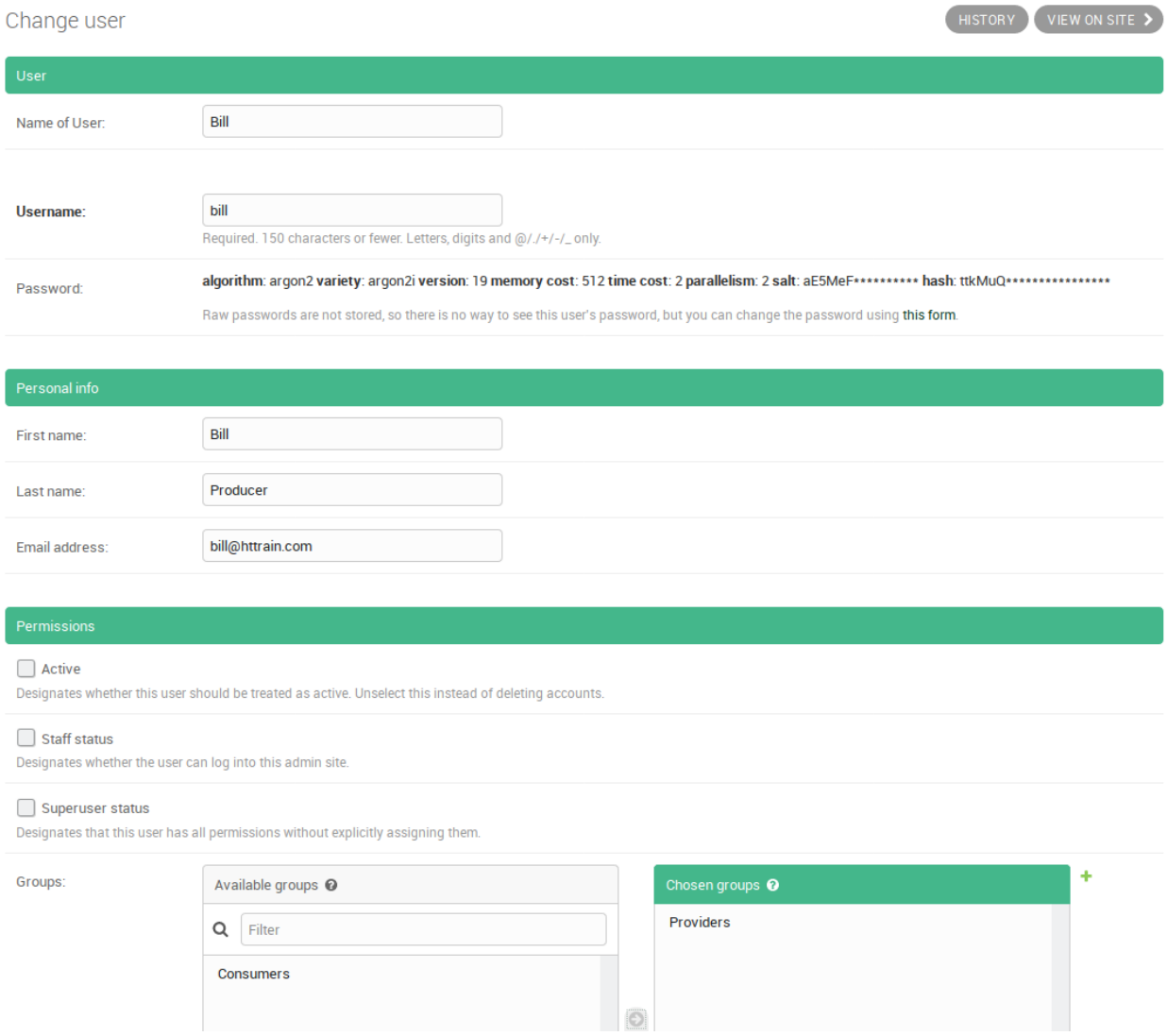


Figure 3: Activation of Provider user “Bill"

2.2 SCENARIO S2: JOINING THE IF USE CASE (USER REGISTRATION)

Actor	**SafeTravel (TrSP):** Travel Applications for smartphone
Target Component/Sub-system/Entity	User Management
Description	This scenario illustrates the registration process for a user that intends to join IF as a “Service Consumer”
Story	Alice is an employee in **SafeTravel** company that develops a smartphone application for ticket search and booking. Alice is responsible to register this transport application to IF. She goes to IF website and selects to register to IF as Service Consumer Role. To create an account, she inserts all the required information related to herself as well as the company, including username, password, type of the company, etc. After successful registration and confirmation of the identity of the registered user, she is then redirected to the Front-end view of **Asset Manager,** which is conceived as the consumer's panel in IF and presents required interfaces to the functionalities available for a service provider such as Asset Discovery.

2.2.1 Tools configuration

The configuration of the Asset Manager to validate S2 is the same which has been used to validate S1.

2.2.2 Validation

The validation of this scenario follows the same steps described for S1. Alice opens the Store application and she is redirected to the login/registration page, as depicted in Figure 4. Once Alice presses “Sign Up”, the admin user is notified via email.

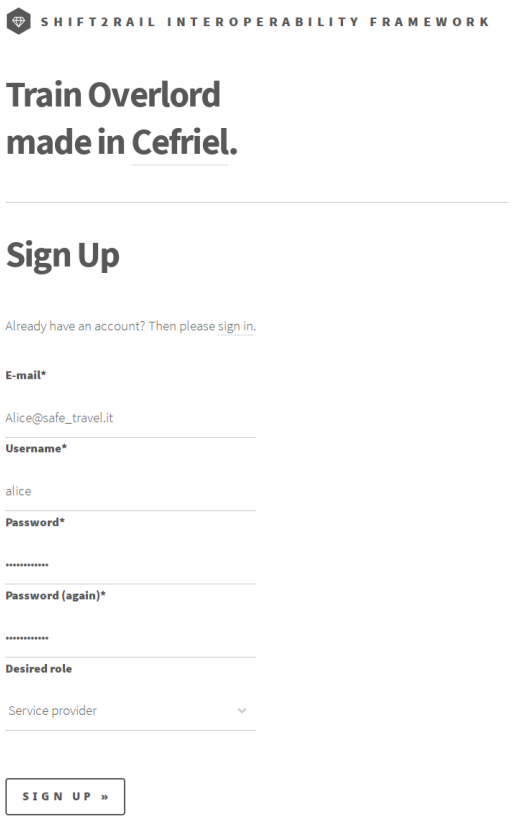


Figure 4: Service provider registration form

The Administrator user opens the Asset Manager administration page and select “Users”, opens the “alice” user and “activates” her allowing logins, as illustrated in Figure 5. Alice is then notified about the activation of his credentials, and she will be able to log into the Store web application.

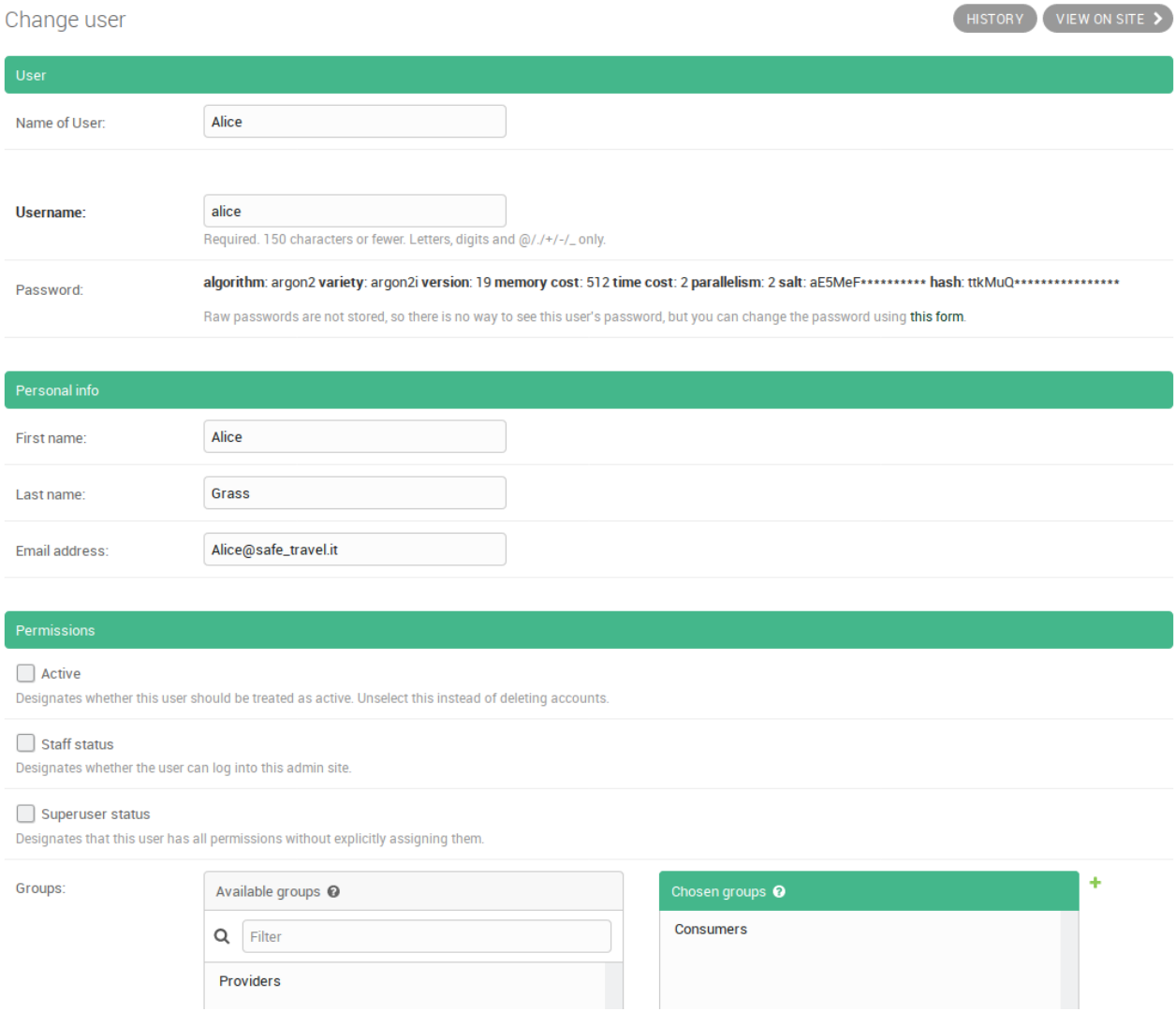


Figure 5: Activation of Consumer user “Alice"

2.3 SCENARIO S3: SERVICE/ASSET DISCOVERY (SIMPLE DISCOVERY)

Actor	NewRail, a rail operator which just joined the IF ecosystem
Target Component/Sub-system/Entity	Asset Manager: Asset Discovery
Description	This scenario demonstrates two basic functionalities of the Asset Manager, namely the possibility to browse the available assets by asset type, and the possibility to perform a faceted search.
Story	NewRail wants to assess the business opportunities of the IF ecosystem, which they just joined. They decide to browse the available assets in the Asset Manager as a first step, exploring the different functions made available by other operators. Then they decide to explore the technical side of the problem, by using the search functionality of the Asset Manager to find out which systems are compatible with the data model and messages specifications that they are using.

2.3.1 Tools configuration

To validate this scenario we configured the Asset Manager with the following asset types:

- Ontologies
- RDF Datasets
- Mappings
- Converters

We published instances for each of those asset types to obtain results during the search phase.

2.3.2 Validation

Validating this scenario requires checking two functionalities: the navigation inside the catalogue and the possibility to perform a search. The Asset Manager Store application let user navigate through the various asset types hosted in the catalogue, and provides for each asset types a page to list all its instances, as depicted in Figure 6.

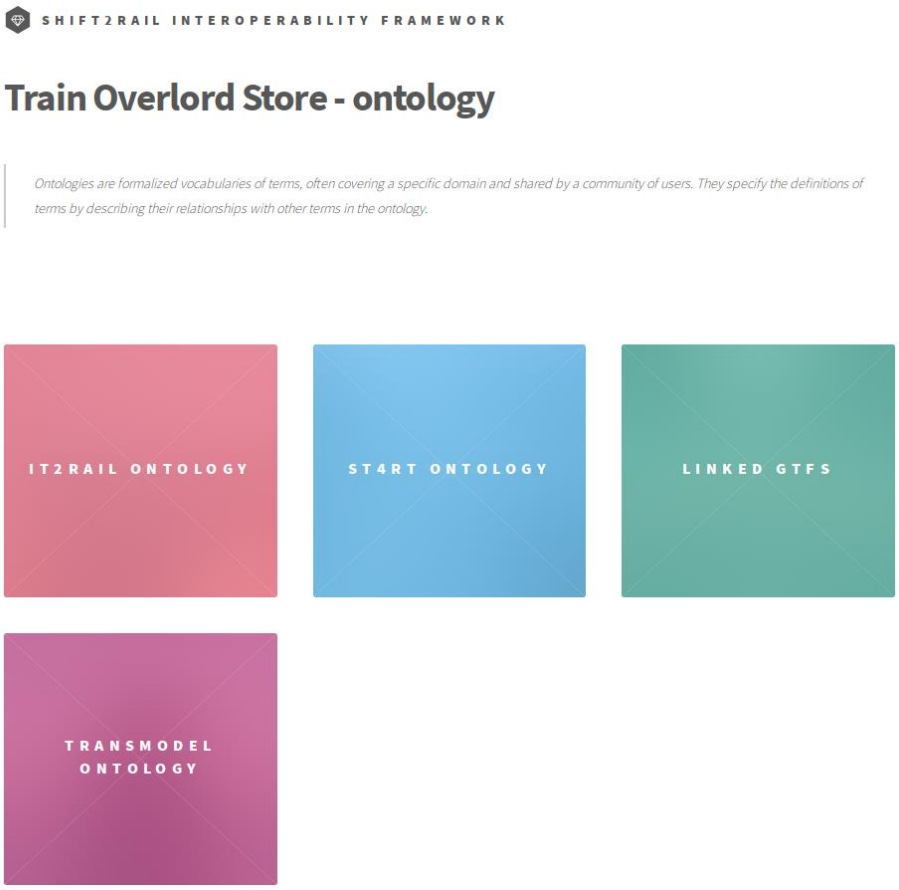


Figure 6: Asset Manager Store listing the available ontologies.

If the user decide to inspect an asset, its metadata is presented, as is shown in Figure 7.

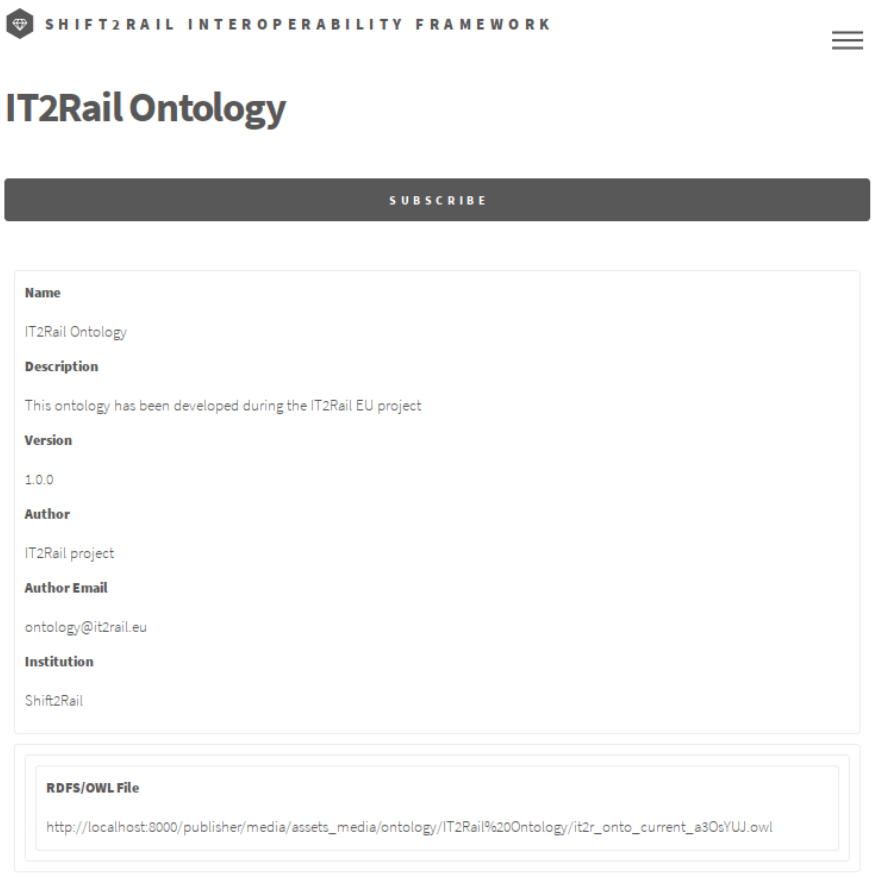


Figure 7: Asset metadata

The user can access the search functionalities by using the menu on the right. As depicted in Figure 8, the user can perform a full-text search, and then refine the visible results by using a set of facets. The C-Rel version of the Asset Manager does not yet allow exploring dependencies between assets. The underlying graph representation of metadata allows performing such operation, which will be implemented in F-Rel.

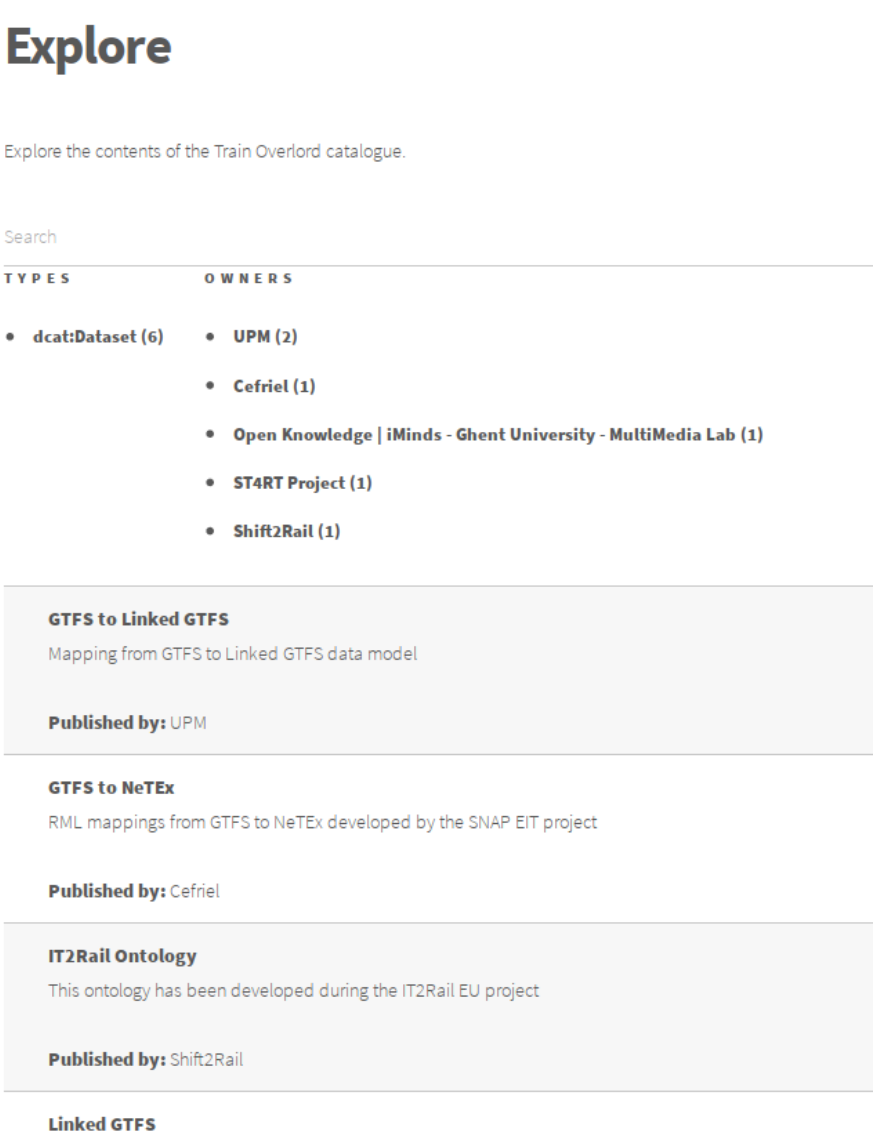


Figure 8: Faceted search in Asset Manager Store

2.4 SCENARIO S4: SERVICE/ASSET DISCOVERY (DISTRIBUTED SPARQL ENDPOINTS)

Actor	B-Com: A Belgian-based TrSP which it is hosted by the (National Access Point) NAP of Belgium S-com: A Spanish-based TSP named “S-com” already hosted by the NAP of Spain. S-com offers travel services within and beyond the Spain boundaries. MyMobility: An Italian company providing transport services in many regions of Europe.
Target Component/Sub-system/Entity	Asset Manager: Distributed SPARQL endpoint
Description	This scenario illustrates querying the SPARQL engine

Story	MyMobility is interested in expanding its routes services across Europe, it wants to discover which service providers are publishing public transport data that may be interesting for them. Once that IF receives a request from MyMobility about catalogues describing data of public transport providers, a distributed SPARQL query process is started. The company requests the list of publishers (transport service providers) of the datasets containing information of the different public means of transport. Since MyMobility is focused on Belgium and Spain to expand its routes services, we treat S-Com and B-Com as two different data sources, whose resources can be combined to find metadata catalogues of the NAP datasets from Belgium and Spain. For this, IF needs to make use of the services provided by B-Com and S-Com, independently of where they are coming from. As such, IF will issue the query to the asset manager distributed SPARQL query engine, which will perform the usual steps of generation of subqueries for each selected source, generating a query plan, rewriting the subqueries considering potential inferences, translating those subqueries and executing them so that the results can be integrated and delivered to the asset manager. To expand its route services in Madrid, MyMobility refines its search after an initial request for catalogues to discover a more detailed information about the Spain datasets comprising descriptions of the stations, bus stops and other infrastructures of S-Com.

2.4.1 Tools configuration

To functionally validate the SPARQL endpoint in scenario S4, our datasets are based on DCAT-AP (a standard widely used in European open data portals) catalogues from Belgium and Spain, and are serialized in Turtle format. The catalogues are published by a Transport Authority Provider and they describe different transportation systems in Belgium and Spain, and they are publicly available 12.

Ontario must be configured with a list of data sources of type SPARQL_Endpoint in a JSON configuration file. Internally, Ontario generates the data schema and its mappings to data sources by means of an abstraction called RDF Molecule Templates. An example of RDF Molecule Templates can be found at 3. An example of JSON configuration file is as follows:

[1] https://github.com/cef-oasis/DCAT-AP/blob/master/TransportDCAT-AP/BELGIUM/TransportDCAT-AP.ttl

[2] https://github.com/cef-oasis/DCAT-AP/blob/master/TransportDCAT-AP/spain/crtm.ttl

[3] https://raw.githubusercontent.com/jatoledo/Ontario/master/configurations/config.json

```
Example: datasources.json
[{
      "name": "spain-rdf",
      "ID": "spain-rdf",
      "url": "http://spain:8890/sparql",
      "params": {},
      "type": "SPARQL_Endpoint",
      "mappings": []
    },
    {
      "name": "belgium-rdf",
      "ID": "belgium-rdf",
      "url": "http://belgium:8890/sparql",
      "params": {},
      "type": "SPARQL_Endpoint",
      "mappings": []
    }
]
```

To validate our scenario inside Ontario, we have expressed two SPARQL queries as follows:

1. The list of publishers publishing datasets:

```
curl -G 'http://localhost:5001/sparql' \
     --data-urlencode query='
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dc: <http://purl.org/dc/terms/>
SELECT distinct ?Publisher WHERE {
?DatasetURI a dcat:Dataset .
?DatasetURI dc:publisher ?Publisher

 }'
```

2. The list of datasets published by Consorcio Regional de Transporte de Madrid:

```
curl -G 'http://localhost:5001/sparql' \
     --data-urlencode query='
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX org: http://www.w3.org/ns/org#

SELECT ?dataset  WHERE {
       ?DatasetURI a dcat:Catalog .
       ?DatasetURI dcat:dataset ?dataset .
       ?DatasetURI dct:publisher ?crtm .
       ?crtm a org:Organization.
       ?crtm foaf:name "Consorcio Regional de Transporte de Madrid"}
```

Such queries have then been integrated into the Asset Manager by means of two Exploration APIs, whose representation inside the Asset Manager publisher is illustrated in Figure 9 and Figure 10.

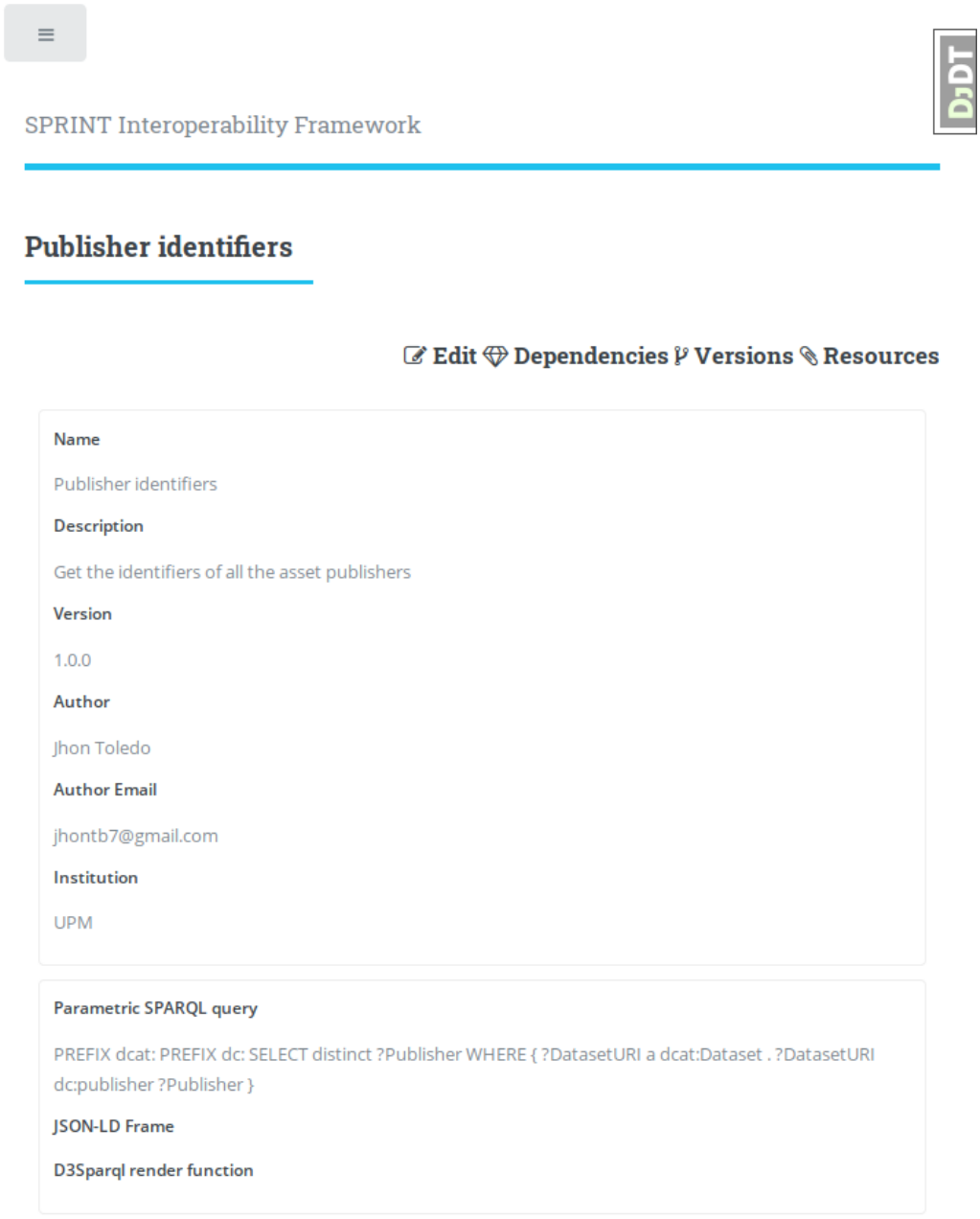


Figure 9: Exploration API to obtain the identifiers of all the asset publishers

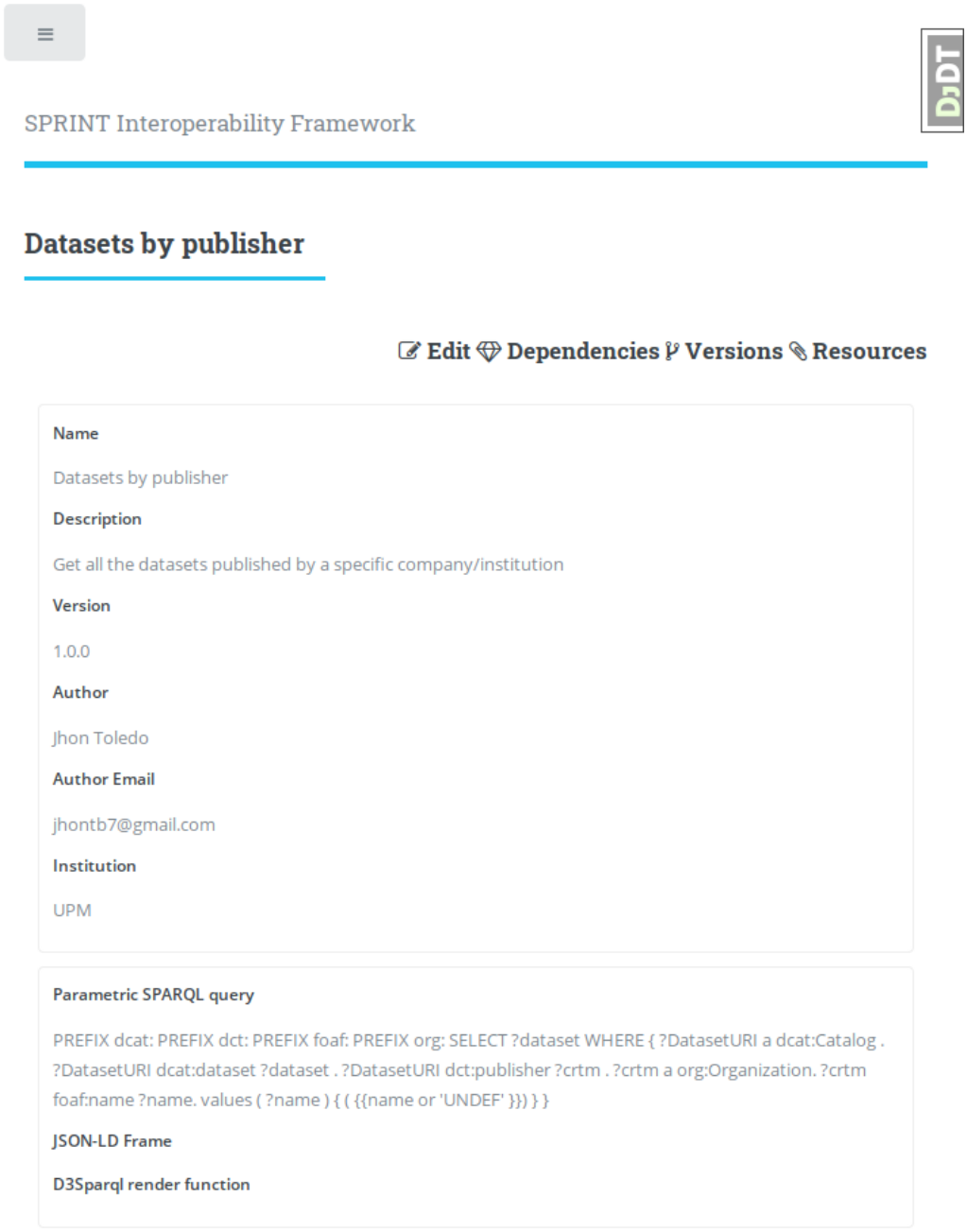


Figure 10: Exploration API to obtain the assets contributed by a specific publisher

Once successfully approved (as all the other asset types, the lifecycle management process requires an explicit approval of the publication of such assets), the Exploration APIs are exposed by the Asset Manager as callable endpoints, and are therefore shown in the Swagger documentation of the Asset Manager API, as shown in Figure 11 and Figure 12. Once published, the Asset Manager becomes the authentication and authorization gateway that checks whether to execute the SPARQL queries implemented using the Distributed SPARQL endpoint, or to deny the access.

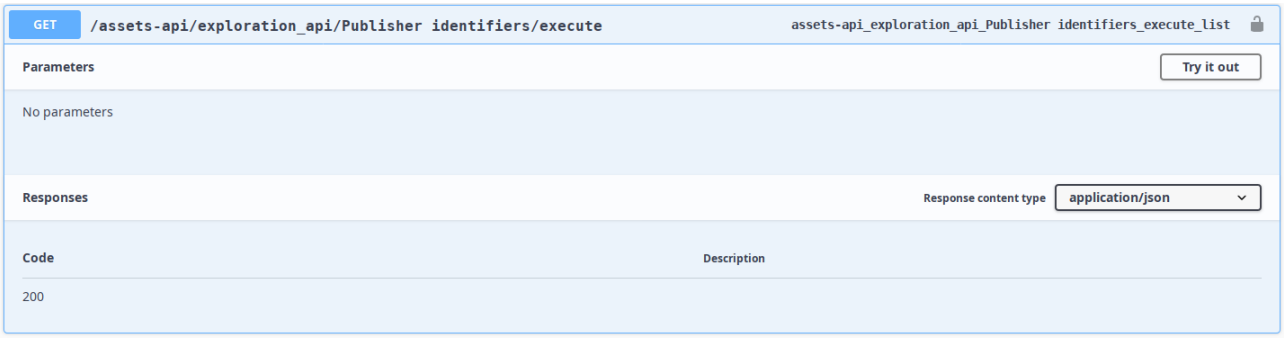


Figure 11: Swagger documentation of the API automatically published by the Asset Manager for SPARQL query 1

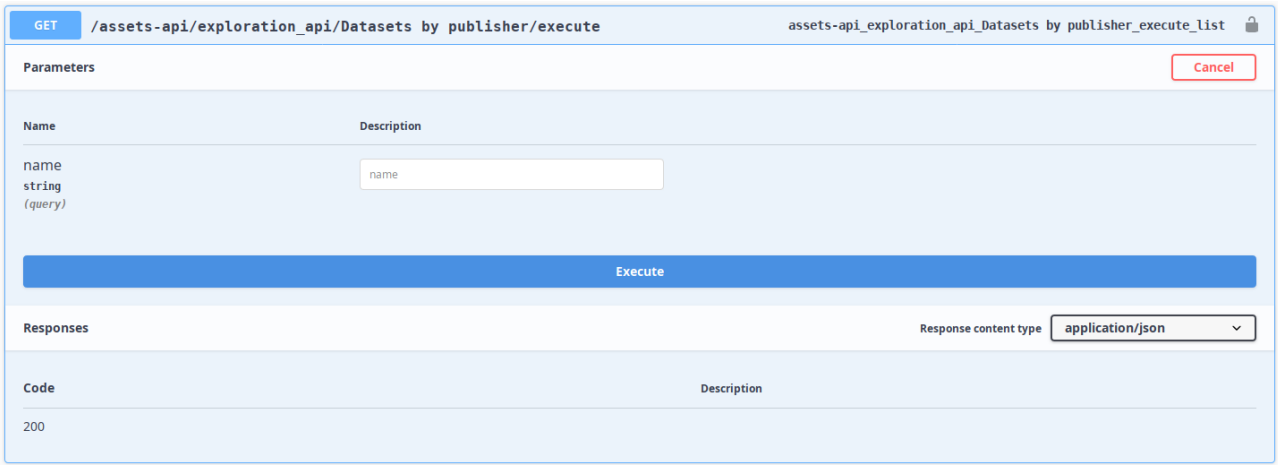


Figure 12: Swagger documentation of the API automatically published by the Asset Manager for SPARQL query 2

2.4.2 Validation

In the C-Rel version of the Asset Manager there is no integration of the results coming from the Exploration API in either the Publisher nor the Store web application. The user can anyway access the functionality using the Swagger interface (which is accessible using the "/assets-api/swagger/" path) or via API tools like Postman or via CLI tools like curl.

The API to obtain the list of datasets published by a specific contributor can be accessed using curl with the following command:

```
curl -X GET "http://localhost:8000/assets-api/exploration_api/Datasets by publisher/execute" -H  "accept: application/json"
```

and the results obtained are the following:

```
{
 "execTime": 0.09533405303955078,
 "firstResult": 0.0922544002532959,
 "querytriples": [

 ],
 "result": [
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/emt-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/emt-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/emt-stops" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/interurban-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/interurbanbus-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/interurbanbus-stops" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-access" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-halls" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-platforms" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/metro-stops" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-access" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-halls" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-platforms" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/train-stations" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-access" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-halls" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-platforms" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/tram-stations" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/urbanbus-gtfs" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/urbanbus-lines" },
  { "dataset": "http://crtm.linkeddata.es/metadata/catalogo/urbanbus-stops" }
 ],
 "totalRows": 27,
 "vars": [
  "dataset"
 ]
}
```


Figure 13: Results of the execution of Exploration API "Datasets by publisher"

The query to obtain the identifiers of the publisher of the datasets can be accessed using

```
curl -X GET "http://localhost:8000/assets-
api/exploration_api/Publisher%20identifiers/execute" -H  "accept:
application/json"
```

and the results obtained are the following:

```
{
 "execTime": 0.10185027122497559,
 "firstResult": 0.0930016040802002,
 "querytriples": [

 ],
 "result": [
  {
   "Publisher": "https://irail.be/#org"
  },
  {
   "Publisher": "http://crtm.linkeddata.es/metadata/organization/crtm"
  }
 ],
 "totalRows": 2,
 "vars": [
  "Publisher"
 ]
}
```


Figure 14: Results of the execution of Exploration API "Publishers identifiers"

2.5 SCENARIO S5: DIRECT ACCESS USE CASE FOR BATCH DATA CONVERSION

Actor	**S_BusTravel (TSP):** A Service Provider
Description	This scenario describes Converter discovery and access for Batch-Data Conversion
Story	**S_BusTravel** is interested to publish a relatively huge set of their data which is in Standard-X to a representation/data model compatible with the target consumers' systems and standards, say Standard-Y. Sara is an IT engineer in **S_BusTravel**. She searches within the IF to find out an X-Y converter. The IF returns two different Converters with the ability to convert Standard A to B. One of them is offered by a famous transport operator **TO** and is accessible as a service**.** The other one is developed by some startup company **Best_Travel** and is provided as a downloadable artifact. Based on the reputation of **TO**, and since the conversion operation has to be performed just once, Sara decides to use the conversion service provided by TO. She opens the TO converter page and retrieves the service URL. The rest of the process is outside of the boundary of IF

2.5.1 Tools configuration

To check the feasibility of this scenario with IF tools, we configured the Asset Manager and we created the “Converter” asset type. A Converter can be either a service or an executable artifact to convert datasets locally in a batch. Moreover, four different ways of accessing the Converter functionalities have been configured:

- Downloadable artifact: the contributor uploads a file inside the Asset Manager, and the consumer can download it.
- Remote artifact: the contributor states where the artifact can be retrieved from.
- External service: the contributor states the endpoint of the service implementing the conversion.
- Automatic building: the contributor asks for automatic synthesis of a Converter, using a basic configuration and reusing already existing assets (ontologies, datasets and mappings).

To instances of the Converter asset type has been then created. Both of them are “Batch converters”, but while the TO Converter metadata contain an “External service” URL, the SC Converter metadata contain a “Downloadable artifact” part which points to an executable JAR.

2.5.2 Validation

Sara connects to the Asset Manager Store application, and after the login she opens the Converter asset type, which contains the list of available assets (as shown in Figure 15).

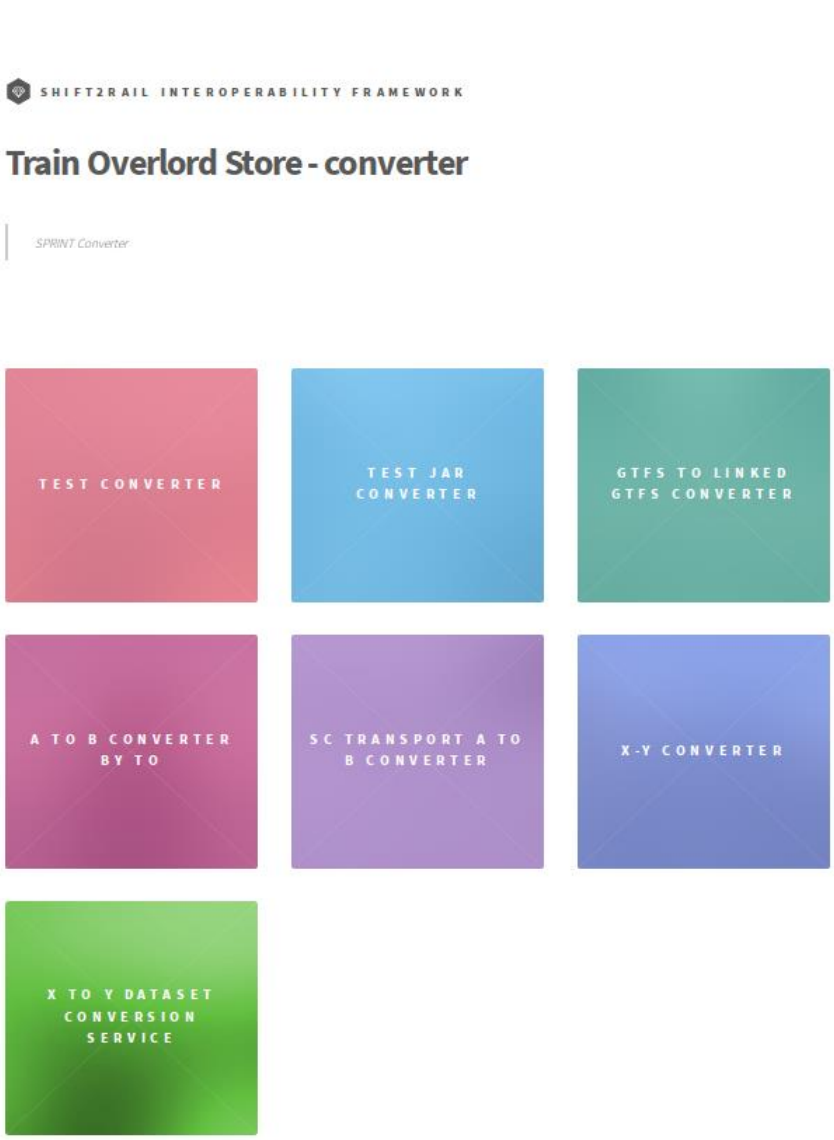


Figure 15: Asset Manager Store: accessing the list of available Converters

She then opens the “X-Y Converter” and the “X to Y dataset conversion service”. The two different metadata presentation pages are shown in . Sara then decides to use the TO-provided service, and therefore accesses the Swagger definition of the service.

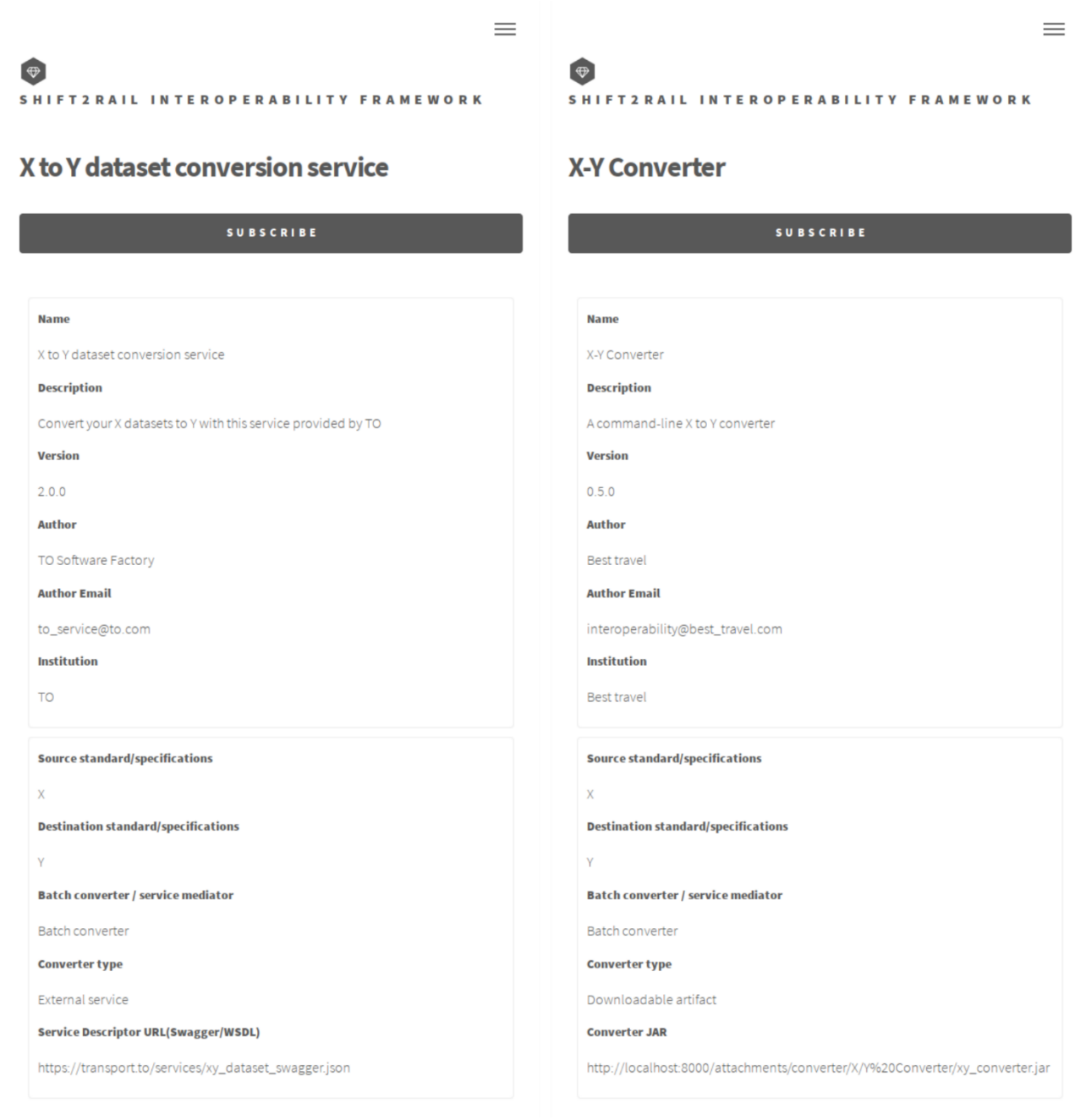


Figure 16: Asset Manager Store: Batch converters metadata

2.6 SCENARIO S6: DIRECT DOWNLOAD USE CASE FOR RUNTIME DATA/MESSAGE CONVERSION

Actor	**BE-Service (TrSP):** an offer building service for land (rail, bus, etc.) travels within central Europe. Its front-end API is used by mobile and web applications and its backend has access to, and, engaged with many train/buses operators in the covered zones.
Description	This scenario describes Converter discovery and deployment (Direct download and then local deployment by consumer) for Runtime Data/Message Conversion
Story	Various smartphone and web applications that are providing means for end-users to search and book tickets for train and bus within Central Europe rely on **BE-Service.** **BE-Service** endpoints receive discovery and booking request from transport applications and return them a list of the available itinerary offered by various transport operator for the requested path. Upon to request of user (through the application), it initiates the booking procedure by forwarding user's request to the ticket provider and completes the booking procedure. Accordingly, the format, specification and standardisation of the booking process differs based on the provider operator. **BE-Service** hence required to convert the source booking request/confirmation format to the target model – and vice versa – instantly at runtime. Bob, the IT engineer in **BE-Service,** searches within IF to find out desired converters. In specific he is looking for A-B converter, A-C converter, and M-C converter. He starts by searching A-B converter. IF returns two different Converters with ability to convert Standard-A to Standard-B. One of them is a service which is offered by a famous transport operator **TO.** The other one is developed by some start-up company **SC** and it exposed as JAR file which could be downloaded and run locally. To make such JAR far accessible, **SC** has uploaded it in the IF repository. Since such message conversion is highly frequent and it is part of a live and runtime transaction, Bob prefers to find some way to integrate such mechanism inside its business logic. Hence, he decides to use the converter provided by **SC.** He downloads the JAR file from IF repository and starts engaging with it accordingly (this process is outside of the boundary of IF).

	Bob initiates another search for A-C converter and repeats the same procedure.

2.6.1 Tools Configuration

To validate this scenario, we used the same asset type configuration for the Converter which has been described in Scenario S5.

In this scenario, we created two Converter assets, one contributed by TO providing a service endpoint, and one contributed by SC providing a downloadable artifact. Both assets have been published and approved by the Administrator user following the lifecycle management process.

2.6.2 Validation

The user accesses the Asset Manager Store and selects the “Converter” asset type. The list of available assets is shown, and the user views the TO-provided and the SC-provided assets, which are shown in Figure 17.

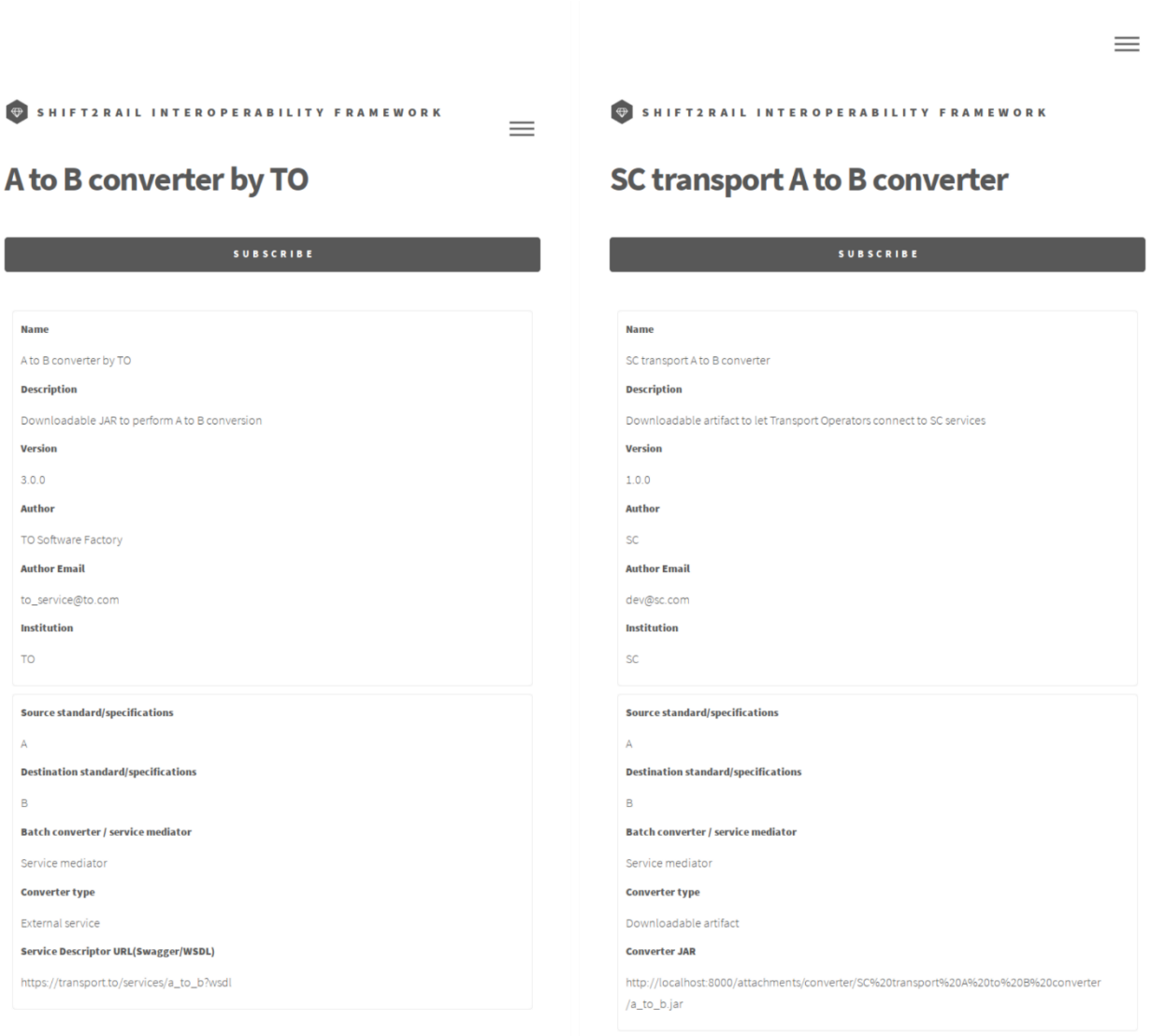


Figure 17: Metadata for SC and TO "A to B" converters as visible from the Asset Manager Store

Since the user prefers downloading the Converter to run it on his own infrastructure, he presses the menu button on the top right corner of the SC-provided asset, and accesses the "Attachments" menu. In the resulting page, he then click on the "Download" button to download the JAR package implementing the conversion, as shown in Figure 18.

Figure 18: Asset Manager Store: downloading a Converter attachment

2.7 SCENARIO S7: AUTOMATED MAPPING PROCESS FOR THE CONVERSION USE CASE

Actor	**Best_Travel (ISA):** A service/application provider
Description	This scenario describes utilisation of Mapping IDE.
Story	According to **Best_Travel** analysis, the Standard-K is becoming more and more popular and widely used that can substitute the other famous by legacy Standard-P. So, they decide to develop a K-P converter and publish it in the market for potential users. To this end, they need to the stablish the “Mapping” between the concepts and terms in both standards which are used to create the annotations as part of conversion mechanism. John is a specialist in the transport domain standardisation in. **Best_Travel** who is part of the team for developing the converter. His role to create the mapping. Mary is another member of the team who is IT engineer and knows about the Mapping utilities of the IF. She has already run the mapping utilities in some local docker container by utilising its docker image from the IF. By initialising the program, it asks the directory address to a structured representation of both source and target standards. After successful uploading standards, it starts the process. When the process is terminated, the output represents a list of the concepts in source format and the suggested equivalent concepts the target standards. John then goes through those suggestions and either confirm or reject them. The program produces the mapping between all the confirmed concepts as final output.

2.7.1 Tools Configuration

The ready-to-use image of the mapping tool can be accessed from docker hub (https://hub.docker.com). The image is not public, so one needs to request access to the image to retreieve it.The command to pull the docker image is the following:

Command: $docker pull safia123/testappv1:mappingtoolv2Onto

To start the mapping process, a user must run the tool providing two files, i.e., the Source and Target stanfdards, as input.

2.7.2 Validation

For given input Source and Target files, the MappingTool suggests mappings between terms of the Source and Target standards. The formats accepted by the tool include OWL, Turtle, XSD, and XML.

The evaluation has been carried out in the following way:

- we chose 2 pairs of ⟨source, target⟩ standars; we call each pair a validation “scenario”;
- for each pair of standards, we ran the Mapping Tool and produced a set of pairs of terms ⟨ $source\_term_i$, $target\_term_j$ ⟩ whose mapping is suggested by the tool;
- for each scenario, we manually checked the mapping suggested by the tool, to evaluate the accuracy obtained for the scenario.

We use the following formula to calculate the overall accuracy of the Mapping Tool for each scenario.

Accuracy percentage = (Number of correct mappings/ Number of feasible mappings)*100

where the meaning of the terms used in the formula is the following:

Number of correct mappings: number of accurately mapped pairs according to the manual evaluation for the corresponding scenario.

Number of feasible mappings: Number of mappings that one could expect the Mapping Tool to find; this value can have slightly different meanings for each scenario, as what is “expected” might vary. For example, for a pair of standard for which a full manual mapping is already available (e.g., the ⟨LinkedGTFS, GTFS⟩ pair), the set of “expected“ mappings the one manually created. However, for a given pair of standards, a full mapping is not always available, hence in this case the notion of “expected” mappings is harder to estimate, and it can simply be taken to be the total number of terms that have a similar concept in the other standard.

The Mapping Tool has been tested against two scenarios, which are described in the rest of this section. The chosen pairs of standards fall in one of two categories: pairs for which an existing manual mapping is available; or pairs for which there was sufficient knowledge to evaluate the reasonableness of the suggested mapping.

Scenario 1: Mapping between LinkedGTFS and GTFS

In Scenario 1 we considered as standards GTFS (General Transit Feed Specification[4]), for which XML representations are available[5] and a semantic version of the standard, called LinkedGTFS[6]. Notice that the two standards have a common root (GTFS), but they are not simply different syntaxes for the same concepts, though of course they share many similarities, hence testing the Mapping Tool on the pair is meaningful. Table 2 summarizes the terms detected in the two standards.

Table 2: Summary of terms in Source (LinkedGTFS) and Target (GTFS) standards

Terminology	Source terms (Linked GTFS)	Target terms (GTFS.xsd)
Unique terms in standard	102	98
Terms remaining after filtering[7]:	101	96
Unique mapped pairs	61	

For the ⟨LinkedGTFS, GTFS⟩ pair a manually-created mapping produced by Cefriel was already available, and it has been used to evaluate the accuracy of the tool.

To evaluate the suggested mappings, the output produced by MappingTool has been compared to mappings produced by Cefriel and validation results are shown below. However, notice that, unlike the mapping suggested by the Mapping Tool, the manually-created mapping did not start from any XML version of the GTFS standard. Hence, there are some discrepancies (discussed below) between the set of terms mapped by Cefriel, and that of the XML file used by the Mapping Tool, which renders some mappings “unfeasible”. These “unfeasible” mappings have een discounted in the evaluation that follows.

[4] https://gtfs.org

[5] https://github.com/CityofSantaMonica/gtfs.bigbluebus.com/blob/master/WebApp/gtfs.xsd

[6] https://github.com/OpenTransport/linked-gtfs

[7] “filtering” is the operation of eliminating terms that are not present in the Google News model vocab list, which is necessary to apply the Word2Vec technique on which the Mapping Tool is based.

Source xml	Target ttl	Decision
Agency	**Agency**	Feasible Correct
ageny_timezone	timeZone	Feasible Inaccurate
agency_lang	language	UnFeasible N
agency_fare_url	fareUrl	Feasible Correct
FareAttribute	**FareClass**	Feasible Inaccurate
price	price	Unfeasible X
currency_type	priceCurrency	Unfeasible X
payment_method	payementMethod	Feasible Correct
transfers	transfers	Feasible Correct
transfer_duration	transferExpiryTime	Feasible Correct
FareRule	**FareRule**	Feasible Correct
fare_id	FareClass	Feasible Correct
route_id	route	Feasible Inaccurate
origin_id	originZone	Feasible Correct
destination_id	destinationZone	Feasible Correct
contains_id	zone	Feasible Correct
FeedInfo	**Feed**	Unfeasible T
feed_publisher_name	publisher	UnFeasible N
feed_lang	language	UnFeasible N
feed_version	vesrion	UnFeasible N
Frequency	**Frequency**	Unfeasible T
trip_id	trip	Feasible Inaccurate
start_time	startTime	Unfeasible T

Figure 19: Mappings suggested by Cefriel Annotations

Figure 19 shows a snippet of the mappings manually produced by Cefriel. There are 57 pairs in total. The column ”decision” shows whether the Mapping Tool also has suggested the same pairs/mappings or not. The meaning of the labels used in column “decision” is shown in Table 3.

Table 3:Description of the meaning labels of column "decision" in Figure 19

Terms	Description
Feasible correct:	The same mapping has been produced by the Mapping Tool
UnFeasible N	Neither of the two terms in the manual mapping is available in the source and target files given as input to the Mapping Tool
Feasible Inaccurate	The mapping tool failed to produce the mapping as expected
UnFeasible T	Of the manually-mapped pair, only the Linked GTFS term exists in the gtfs.ttl input file, whereas the GTFS term does not exist in the gtfs.xsd file
UnFeasible X	Of the manually-mapped pair, only the GTFS term exists in the gtfs.xsd input file, whereas the Linked GTFS term does not exist in the gtfs.ttl file

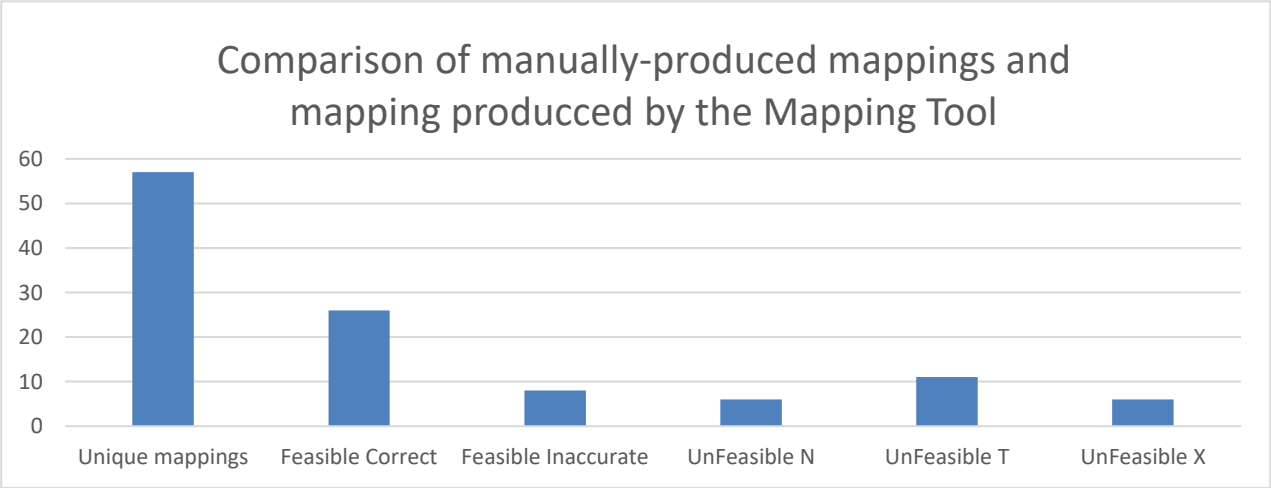


Figure 20:Comparison of manually-created and automatically-created mappings

A comparison of the output produced by the Mapping Tool with the mappings manually created by Cefriel produces the results shown in Figure 20.

An analysis of the results shows that, out of 57 manually-created mappings (a snippet of which is shown in Figure 19), in 7 cases neither terms were part of the input files (hence, they are categorized as "Unfeasible N"); in 11 cases the mapping included a term that is present in the Turtle file, but the other term is not in the XML file (hence, these pairs are labeled as "UnFeasible T"); in 6 cases the mapping included a terms from the XML file, but no term from the Turtle file (these pairs are labled as "Unfeasible X"). Figure 21 shows the accuracy of the MappingTool considering only feasible terms. More precisely, wihle the number of total manually-mapped pairs is 57, 23 of them include unfeasible terms. Hence, the number of feasible mappings is 57-23= 34, which is shown in Figure 21 as "Feasible 60%". This means that, out of 57 total unique manually-created pairs, 60% were found feasible for automatic mapping. Of these 34 pairs, 26 were accurately mapped by the Mapping

Tool, whereas 8 pairs are miseed. Hence, we can conclude that for the given scenario the accuracy is (26/34)*100= 76%, as shown in Figure 21 throuhg label “Correct 76%”.

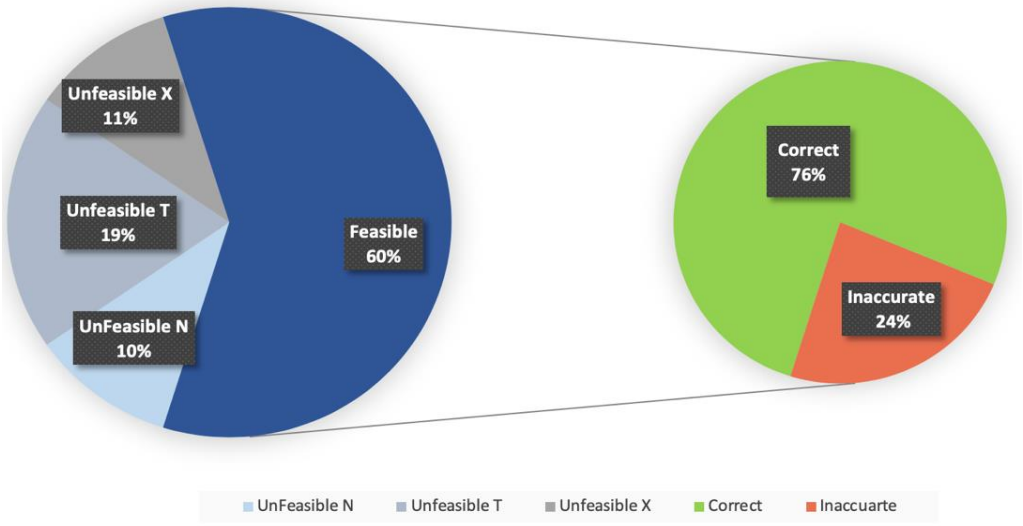


Figure 21: Mapping Tool Accuracy

Scenario 2: Mapping Between FSM standard and IT2Rail Ontology.

In the second analyzed scenario we considered the ontology developed within the IT2Rail project[8], and the FSM standard that was used as a case study in the ST4RT project[9]. In particular, we chose a subset of the FSM standard, related to the infrastructure, that seemed to include a number of concepts that should also be available in the IT2Rail ontology. In addition, the terms of the FSM standards are sufficiently well commented that our confidence in the accuracy of our manual evaluation the mappings is good. Table 4 summarizes the sets of terms retrieved from the source and target standards.

Table 4: Summary of terms in Source (FSM) and Target (IT2Rail) standards

Terminology	Source terms FSM	Target terms IT2RAIL
Unique terms in source	83	555
Terms remaining after filtering	82	527
Unique mapped pairs	72	

Figure 22 shows a snippet of the manual evaluation of the mappings suggested by the Mapping Tool when run on the FSM and IT2Rail standards. While evaluating the suggested mappings, we realized that, for some concepts in the source standard (FSM), no equivalent concept was actually available

[8] http://www.it2rail.eu

[9] http://www.st4rt.eu

in the target standard (IT2Rail). We label pairs including these concepts as “Unfeasible X” and color them in red in Figure 22. In particular, we found 40 unfeasible pairs. Moreover, we labeled as “Unfeasible Near Miss” 7 pairs (highlighted in orange color in Figure 22), since a concept corresponding to the FSM term does not seem to exist in the IT2Rail ontology, though a similar one could be found. Table 5 lists the labels used in Figure 22. An example of these “near misses” are the terms in the FSM standard that represent element IDs, which seem to be mappable to the “hasId” property in the IT2Rail ontology, though the definition of the “hasId” property is not present in all concepts in the IT2Rail ontology.

Source	Suggestion	Decision
Service Mode Id	is In Mode Of Transport	UnFeasible X
Route Type Id	is For Route Link	Feasible Inaccurate
Train Path Type Id	has PRM Type Type	UnFeasible X
Service Availability Id	Web Service Descriptor	UnFeasible X
Origin	has Origin	Feasible Correct
Stop Place	Stop Place	Feasible Correct
Destination	has Destination	Feasible Correct
Connection	has Version	UnFeasible X
Vehicle	has Vehicle Operator	UnFeasible X
Service Brand Id	has Ancillary Service	UnFeasible X
Onboard Service Category Id	has Smart Device Requirements	UnFeasible X
Segment Id Ref	Segment	UnFeasible X
Segment Ref	has Ref To Token	UnFeasible X
Passenger Id Ref	has Ref To Travel Expert	UnFeasible X
Passenger Ref	has Ref To Token	UnFeasible X
Place Type Id	has PRM Type Type	UnFeasible X
Coach Id	Coach Station	UnFeasible X
Compartment Id	has Ref To Payment Module ID	UnFeasible X
Place Id	Stop Place Code	UnFeasible X
Width Pos	Means Of Transport Type	UnFeasible X
Length Pos	has Ref To Access System Item	UnFeasible X
Compartment Width Pos	PRM Type	UnFeasible X
Neighbour Passenger Id Refs List	has Ref To Stop Place Code	UnFeasible Near Miss
Neighbour Passenger Refs List	has Ref To Passenger	Feasible Correct
Stop Place Name	Stop Place Code	UnFeasible X
Latitude	has Latitude	Feasible Correct
Longitude	has Longitude	Feasible Correct
SRS Name	has Product Name	UnFeasible X
Stop Area Id	has Ref To Stop Place	UnFeasible Near Miss
Name	has Product Name	UnFeasible Near Miss
Vehicle Access Area Type Id	has Message Type	UnFeasible Near Miss
Boarding Position Id	Seat Type	UnFeasible Near Miss

Figure 22: FSM to IT2Rail Mappings provided by Mapping Tool

Table 5: Description of labels used in column "decision" in Figure 22

Decision	Description
Unfeasible X	The term in the FSM standard does not have a corresponding concept in the IT2Rail ontology.
Feasible Correct	The mapping suggested by the Mapping Tool is correct.
Feasible Inaccurate	The mapping suggested by the Mapping Tool is incorrect, though a correct one seems to exist.
Unfeasible Near Miss	The term in the FSM standard does not seem have a corresponding concept in the IT2Rail ontology, though a similar one seems to exist.

Figure 23 shows a summary of the findings of the manual evaluation of the mappings suggested by the Mapping Tool for the FSM and IT2Rail standards.

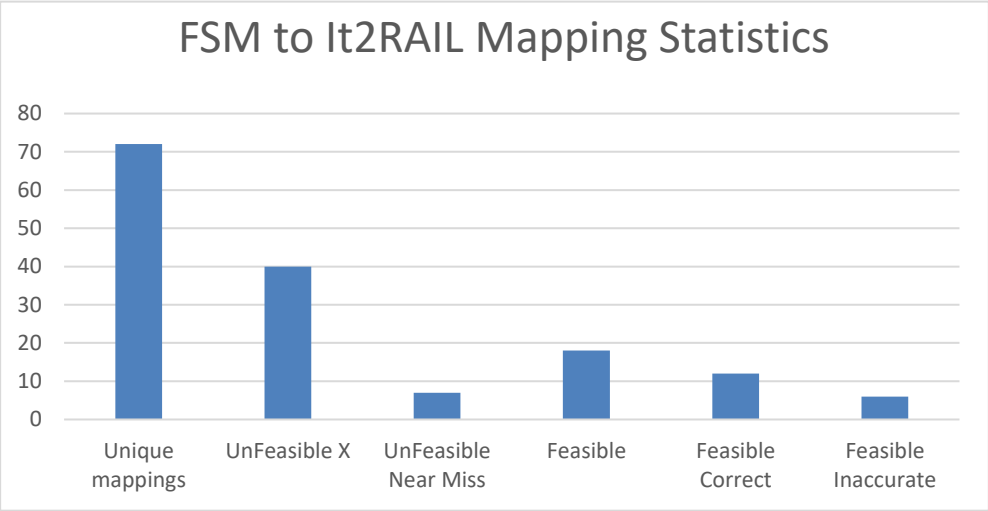


Figure 23: FSM to IT2Rail Mapping Statistics

As mentioned in Table 4 there are only 83 terms in the considered subset of the FSM standard and 555 terms in the IT2Rail ontology. Hence, there are many terms in the IT2Rail ontology that are not available in the FSM standard, and which are then disregarded in the counts. Figure 24 graphically shows the estimated accuracy of the tool on the FSM and IT2Rail standards.

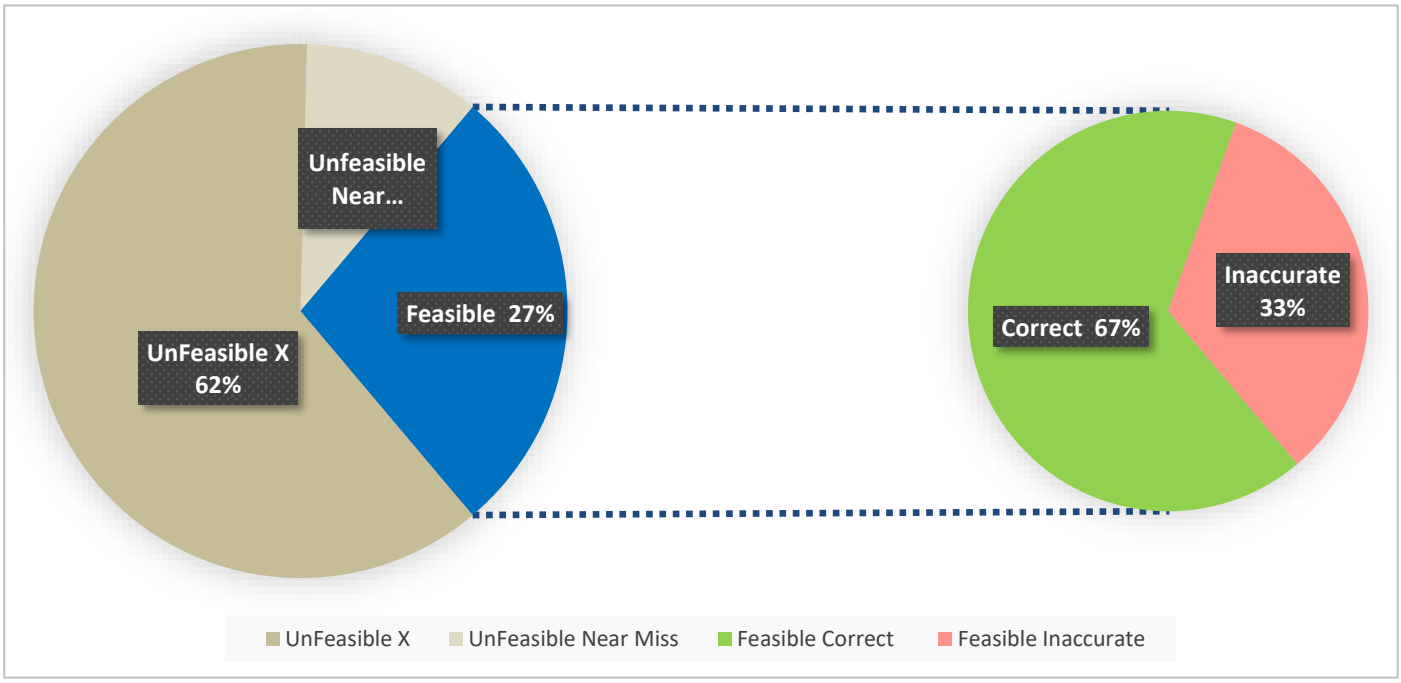


Figure 24: Mapping Tool Accuracy in Percentage

Out of 72 unique pairs, only 18 were found to be feasible, as shown in Figure 24 through label "Feasible 27%". Of those Feasible pairs, 12 pairs were found to be accurately mapped by the Mapping Tool, and 6 were considered to be inaccurate (but feasible). Hence, the estimated accuracy is (12/18)*100= 67%, as graphically shown in the right part of Figure 24.

Summary

The accuracies estimated for the two scenarios presented above the the following

Scenario 1: 76%

Scenario 2: 67%

We compute the average accuracy as follows:

Average Accuracy = (76 + 67)/2 = 71.5%

To conclude, we estimate that the overall average accuracy of the Mapping Tool on the considered scenarios is 72%.

2.8 SCENARIO S8: AUTOMATIC CONVERTER BUILDING USE CASE

Actor	**N-rail:** a rail *TSP* which just joined the Shift2Rail ecosystem. **Y-bus and X-bus**: bus *TSPs* already part of the Shift2Rail ecosystem.
Description	A new operator is interested in establishing a new business by communicating with other operators who already joined the IF ecosystem. Since those operators are compliant with the Shift2Rail Ontology, he just needs to provide the mapping between the messages used by his IT systems and the reference ontology. The Asset Manager will then be able to assemble a Converter, composing the different mapping and the required ontologies and data sets. Such Converter will be then used by the operator to effectively connect his system to the ones provided by the other operators.
Story	Y-bus services joined the S2R ecosystem and contributed a Converter to let its clients interact with X-bus, an allied bus operator. It does so by providing a mapping which “lifts” its own data model to the Shift2Rail ontology, and also a mapping which “lowers” instances of the S2R ontology to the X-bus data model.

2.8.1 Tools Configuration

The automatic building features leverages on Jenkins to perform the operation. Through the Asset Manager admin interface, it is possible to create Jenkins jobs and to connect them to Asset Types. If the lifecycle management process is configured to triggering them, such jobs are then executed.

In the case of Converter artifact building, we first created the “generate converter” job via the admin interface. As depicted in Figure 25, such job is composed by two main parts, namely the job configuration and the related files. Jenkins is a CI/CD solution that can either run as part of the Asset Manager, or indepentently if already installed by the company running the IF node. The job configuration is divided into “stages”, and each stage is composed by instructions. The first stage (“Prepare”) collects all the required resources from the Asset Manager itself. In the case of Converters, three main items are fetched from the Asset Manager:

- the Converter metadata;
- the Chimera library, compiled as a single JAR file containing all the dependencies required to run the Converter;
- utility files which are used to analyse the Converter metadata and create the Converter configuration, using a configuration template.

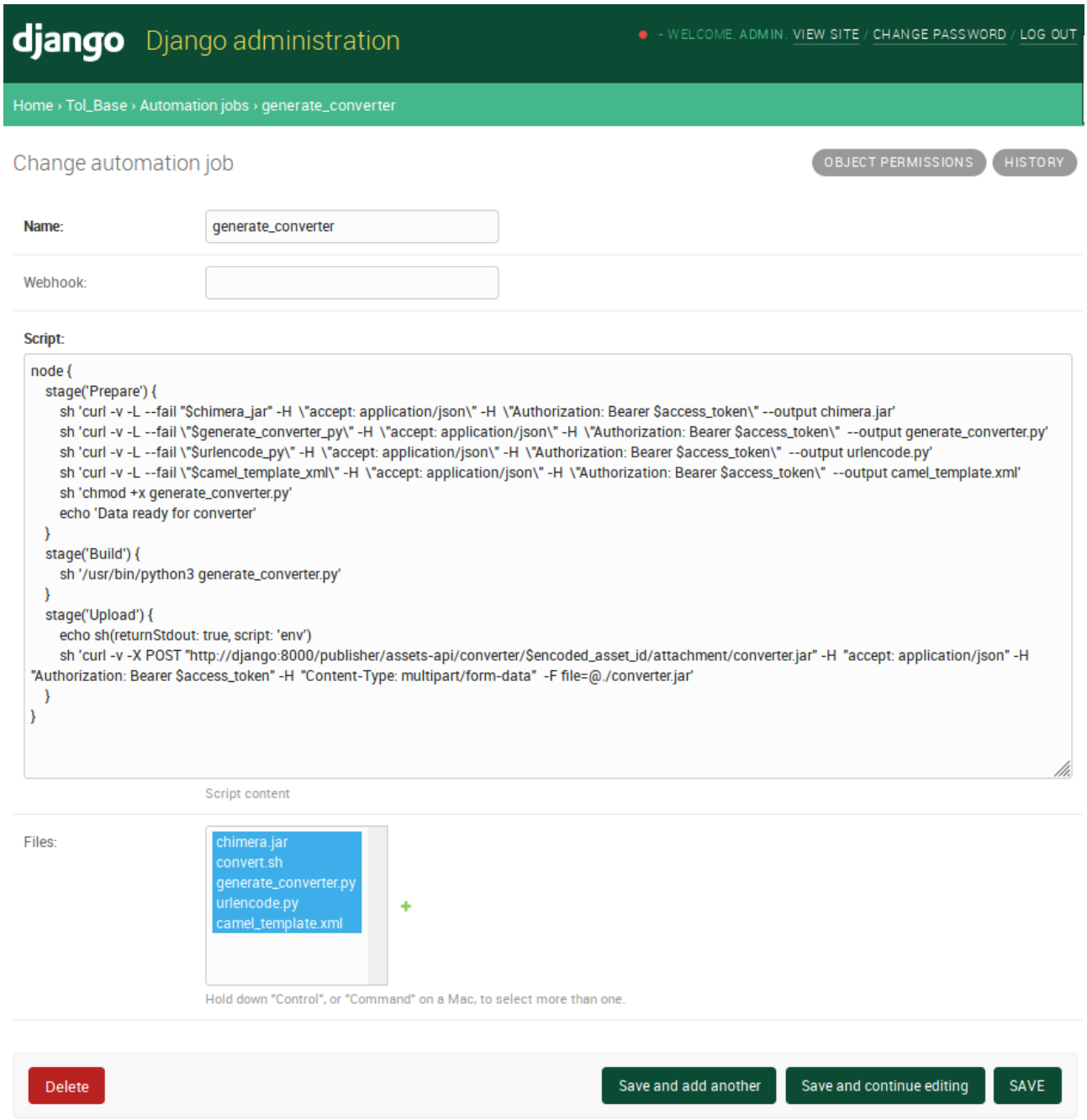


Figure 25: Jenkins job configuration

Once such Automation Job is saved, the job is sent to Jenkins, which will then be able to execute it (as illustrated in Figure 26,showing the administration interface of the Jenkins server).

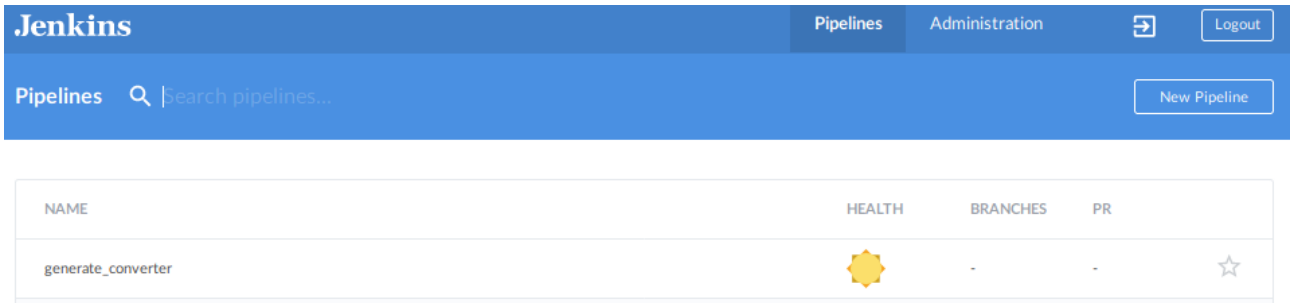


Figure 26: Jenkins administration interface

The lifecycle process for this test was simplified wrt. the default process requiring an explicit consent to publication. As depicted in Figure 27, the BPMN process directly triggers the execution of the Jenkins jobs attached to the “Converter” asset type (in this case, “generate converter” job).

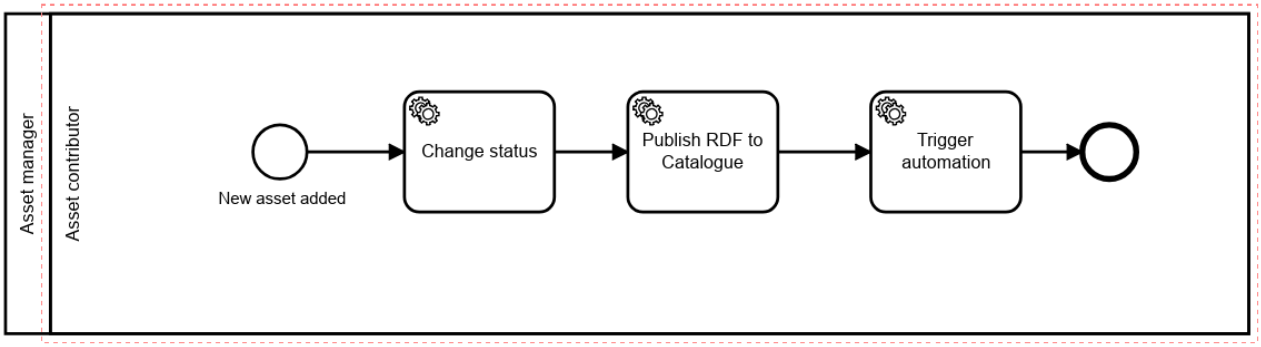


Figure 27: Converter lifecycle

2.8.2 Validation

To validate the automatic building of artifacts, we started creating two Mapping assets, which are shown in Figure 28. The former one allows us to state that there is a mapping from the Y-bus system to the Shift2Rail ontology, which is based on RML and is uploaded as an attachment in the Asset Manager. The second one states the existence of a mapping from the Shift2Rail ontology to the X-Bus data model, and that such mapping is based on an Apache Velocity template containing SPARQL queries (as supported by SPRINT Chimera converter framework).

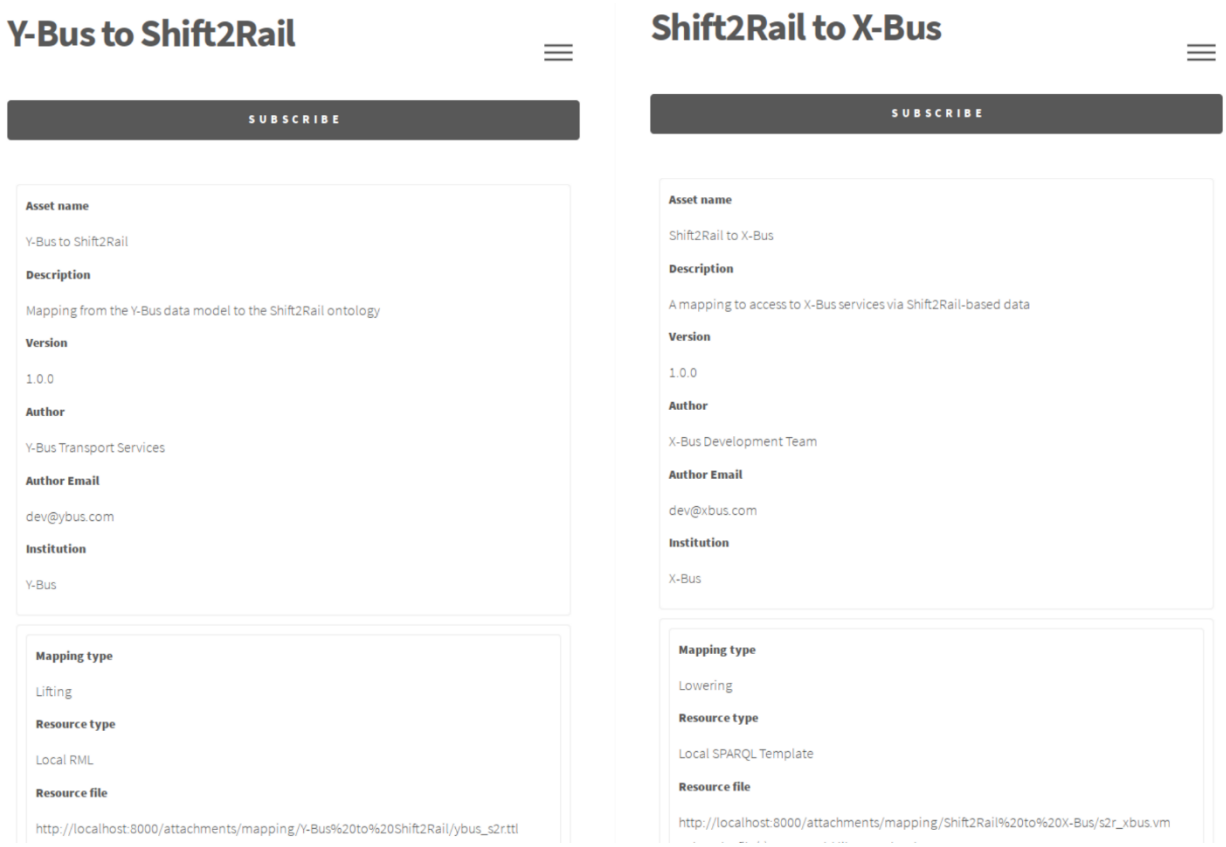


Figure 28: Mappings used in Scenario S8

We then created the Converter asset, inserting as lifting mapping “Y-Bus to Shift2Rail” and as a lowering mapping “Shift2Rail to X-Bus”, as illustrated in Figure 29.

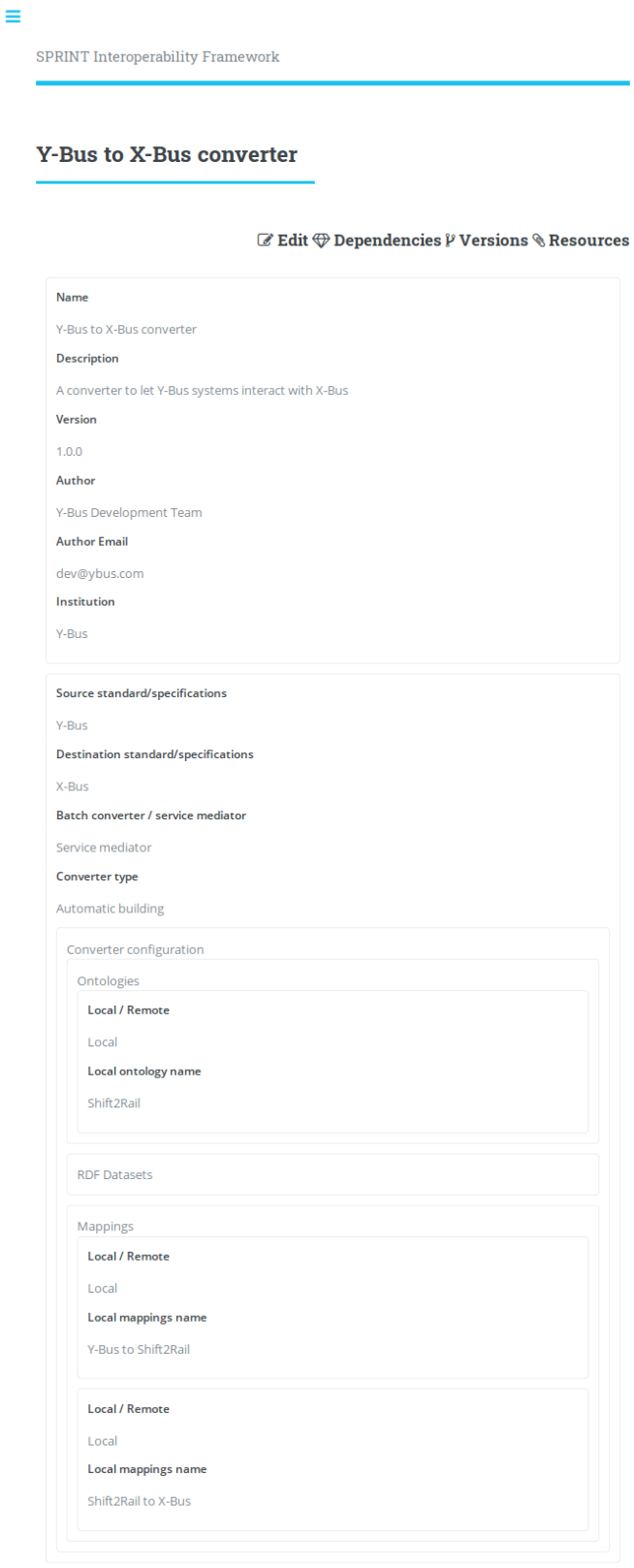


Figure 29: S10 Converter description

When the Converter is published, the Jenkins automation job described in the Configuration section is automatically started. Upon completion, the end user is able to download the Converter JAR file. To do that, he just needs to open the Store application, locate the “Y-Bus to X-Bus converter” asset, open its “Attachments” menu (which has been described in the previous scenarios) and download the file.

2.9 SCENARIO S9: FAST ADAPTATION TO PEAKS USE CASE

Actor	**BE-Service:** Booking Engine for land (rail, bus, etc.) travels within central parts of Europe. Its front-end API is used by mobile and web applications (say T-A-1 to T-A-10) and its back-end has access to, and, engaged with many train/bus operators (say T- O-1 to T-O-20) in the covered zones.
Description	The infrastructure managing the converters deployed by BE-Service to interact with its partner operators need to dynamically adapt to the load. BE-Service needs to quickly replicate Converters, possibly in a cloud environment, to adapt the infrastructure and avoid denial of service.
Story	One of the cities covered by BE-Service is hosting a huge music event, and BE-Service expects a surge of booking request. BE-Service, therefore, needs to cope with two different scenarios: prepare for the first wave of requests to reach the city, and then to cope with mass requests to reach the music event before its start and to reach the homes and hotels after its end.

2.9.1 Tools Configuration

To validate this scenario, we first created a Docker compose descriptor which is able to start two services, namely a load balancer (an Nginx instance configured to act as a reverse proxy) and the Converter microservice. The descriptor looks as follows:

```
version: '3.4'

services:

  converter:

    build:

      context: .

      dockerfile: ./Dockerfile

    image: converter


  nginx:

    image: nginx:latest
```

```
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
    depends_on:
      - converter
    ports:
      - "4000:4000"
```

The Nginx configuration used to obtain a reverse proxy is :

```
user  nginx;
events {  worker_connections   1000;  }
http {
        server {
              listen 4000;
              location / {  proxy_pass  http://converter:8888;  }
        }
}
```

This will configure Nginx to forward the request from port 4000 to http://converter:8888. This will then be resolved by Docker's embedded DNS server, which will use a round robin implementation to resolve the DNS requests based on the service name and distribute them to the Docker containers.

After checking this set up of the scalable converter, we then integrated the creation of all those configuration files in the same Jenkins job which is used to automatically create Converters which has been explained in Scenario S8. When the user publishes a new Converter and asks for the automatic building of the artifact, the result of the Jenkins job execution which is triggered by the lifecycle management process is therefore the publication on the Asset Manager of two attachments, namely the Converter JAR and a zip file containing the JAR file plus all the required configuration files which can be used to start a scalable converter.

2.9.2 Validation

The process described in Scenario S8 creates both the downloadable artifact and the Docker compose configuration. To access it, the user must open the Converter page, and access the "Attachment" page from the menu. The user must then download the "docker-compose.zip" attachment. Inside it he will find two files:

- a Dockerfile to build the microservice image;
- a Docker Compose configuration stating the initial number of instances.

To build and start the containers, the user will use the command

```
docker-compose up
```

To adapt the number of instances of the Converter to the traffic, the user will then use the command

```
docker-compose scale converter=NUM
```

where NUM is the desired number of instances of the converter to be run. The load balancer of provided with the Docker Compose environment will take care of routing the requests to the available Converter instances.

The C-Rel version of this scenario focuses on Docker compose, which is a single-node orchestrator. In F-Rel, we will investigate further on the issue, investigating how to scale Converters using a cloud orchestrator like Kubernetes.

2.10 SCENARIO S10: SPECIAL PURPOSE ASSET DISCOVERY PACKAGE: RESOLVER

Actor	Semantic Converter
Description	Special-purpose Asset Discovery components, or Resolvers, are packaged as deployable units and used to perform discovery/retrieve of specific categories of resources such a Locations or Travel Expert services. These Resolvers can be deployed equally internally to the Interoperability Framework, or in any external runtime environment, e.g. at the Travel Service Provider. A Collection of them are already available since their development in project IT2Rail, i.e. Location Resolver, Travel Expert Resolver, Location Identification, NeTEX Stop Place provider, Navitia Decoder. In scenario S10 of the SPRINT project a new specialized special-purpose Resolver, **Ontology Mappings Resolver**, is used to provide Ontology RML mappings and SPARQL Queries to semantic Converters to drive the lifting and lowering phases of a conversion instance. The Ontology Mappings Resolver is deployed within the Asset Manager or externally to it, and it exposes a RESTful web service that is called by Converters to obtain the specific mapppins an d/or queries that are needed to complete a conversion. A REST client bound to the Ontology Mappings Resolver's endpoint is integrated during Automatic Converter Building, decribed in Scenario 8, and used at lifting and lowering time to obtain the ontology resources needed. In this way RML mappings and SPARQL queries are externalized, residing on the Asset Manager, and Convertes that need them do not need to be rebuild when they change.
Story	Requesting Actor, a semantic Converter automatically built as described in Scenario 8 1. In the lifting fase (lowering phase) the Converter calls the REST direct:rmlmapping (direct:loweringqueries) endpoints, respectively, which instantiates an HTTP Client to call the Ontology Mappings Resolver service 2. The Resolver validates and analyses the call 3. If the query is valid, it is passed to the Asset Discovery component for processing. The Process Request activity may use the Distributed SPARQL endpoint to access the specific requested assets: RML mappings (SPARQL queries) 4. The Resolver then builds a response to be returned to the requestor Actor.

The following diagram illustrates the relationship between the Ontology Mappings Resolver and a Converter

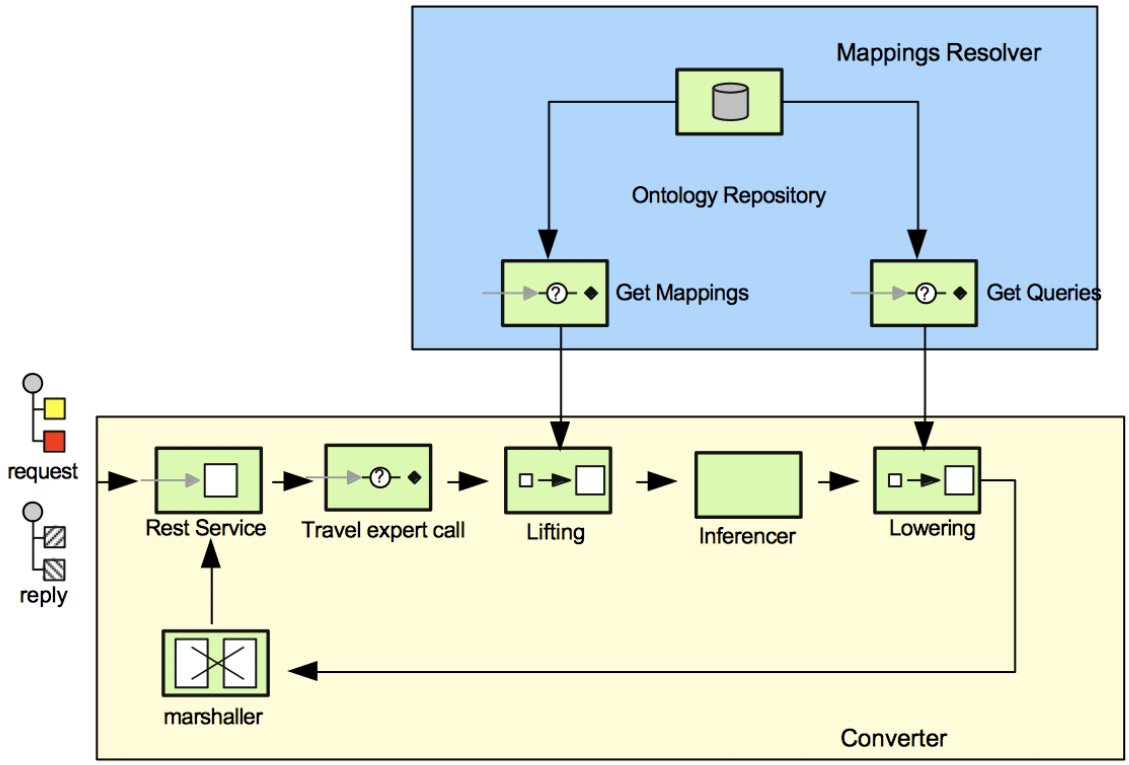


Figure 30: Ontology Mappings Resolver

2.10.1 Tools Configuration

The Ontology Mappings Resolver is implemented as a SpringBoot Apache Camel application whose configuration parameters are held in an application.properties file. The following is an example of the configuration specific for mappings and queries needed by the Hacon converter.

```
camel.springboot.name    =  OntologyMappingsResolver
endpoints.enabled        =  true
endpoints.health.enabled =  true
server.address           =  0.0.0.0
management.address       =  0.0.0.0
management.port          =  8081
camel.springboot.tracing =  true
trips.rml.resource       =  VBB/VBBTriasTripStructure3.ttl
fares.rml.resource       =  VBB/VBBTriasFaresStructure.ttl
functions.rml.resource   =  VBB/VBBfunctions.ttl
queries.rml.resource     =  VBB/VBBTripLoweringQueries.ttl
```

The Conveter configuration generated during Automatic Converter Building (scenario 8) has specific entries to allow the Lifting and Lowering phases to invoke the Ontology Mappings Resolver, as in the following example for the VBBTrips2TriasConverter

```
camel.springboot.name    = VBBTrips2TriasConverter
camel.springboot.tracing = true
endpoints.enabled          = true
endpoints.health.enabled = true
server.address         = 0.0.0.0
management.address       = 0.0.0.0
management.port          = 8082
trips.rml.service        = rest:get:rml/vbb/trips?host=x.y.z
fares.rml.service        = rest:get:rml/vbb/fares?host= x.y.z
queries.rml.service      = rest:get:rml/vbb/queries?host=x.y.z
functions.rml.service    = rest:get:rml/vbb/functions?host=x.y.z
hacon.service.uri        = http://fahrinfo.vbb.de
hacon.service.path       = restproxy/trip
```

The lines in bold in the configuration direct the Converter to invoke the corresponding REST services exposed by the Ontology Mappings Resolver.

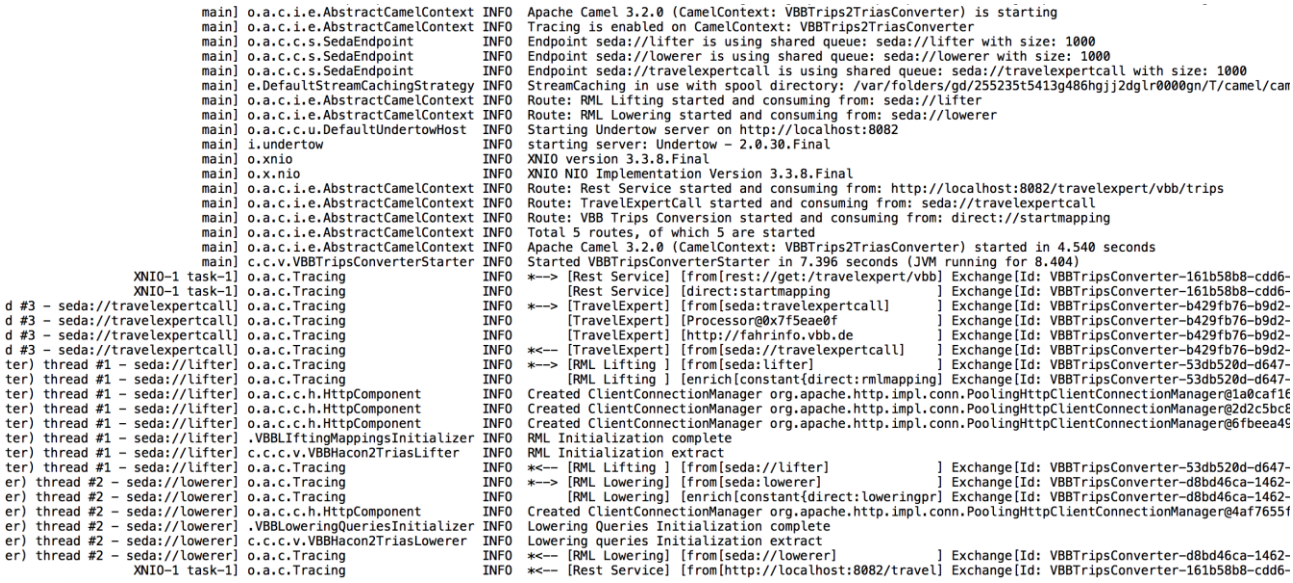
2.10.2 Monitoring and Validation

The screen-shot below shows a trace generated by the Ontology Mapper Resolver at startup, exposing four REST web services: Trips mappings provider, Fares Mapping Provider, Functions mappings provider and Lowering queries provider. It also shows (lines marked by o.a.c.Tracing a series of requests made by a Converter to those REST endpoints:

```
          main] o.a.c.i.e.AbstractCamelContext INFO  Apache Camel 3.2.0 (CamelContext: OntologyMappingsResolver) is starting
          main] o.a.c.i.e.AbstractCamelContext INFO  Tracing is enabled on CamelContext: OntologyMappingsResolver
          main] o.a.c.i.e.AbstractCamelContext INFO  StreamCaching is not in use. If using streams then its recommended to enable stream caching. See more de
          main] o.a.c.c.u.DefaultUndertowHost  INFO  Starting Undertow server on http://0.0.0.0:8081
          main] i.undertow                     INFO  starting server: Undertow - 2.0.30.Final
          main] o.xnio                         INFO  XNIO version 3.3.8.Final
          main] o.x.nio                        INFO  XNIO NIO Implementation Version 3.3.8.Final
          main] o.a.c.i.e.AbstractCamelContext INFO  Route: Trips mappings provider started and consuming from: http://0.0.0.0:8081/rml/vbb/trips
          main] o.a.c.i.e.AbstractCamelContext INFO  Route: Fares mappings provider started and consuming from: http://0.0.0.0:8081/rml/vbb/fares
          main] o.a.c.i.e.AbstractCamelContext INFO  Route: Functions mappings provider started and consuming from: http://0.0.0.0:8081/rml/vbb/functions
          main] o.a.c.i.e.AbstractCamelContext INFO  Route: Lowering queries provider started and consuming from: http://0.0.0.0:8081/rml/vbb/queries
          main] o.a.c.i.e.AbstractCamelContext INFO  Total 4 routes, of which 4 are started
          main] o.a.c.i.e.AbstractCamelContext INFO  Apache Camel 3.2.0 (CamelContext: OntologyMappingsResolver) started in 0.681 seconds
          main] r.ChimeraRmlServiceApplication INFO  Started ChimeraRmlServiceApplication in 3.374 seconds (JVM running for 4.245)
XNIO-1 task-1] o.a.c.Tracing                   INFO  *--> [Trips mappin] [from[rest://get:/rml/vbb/trips?co] Exchange[Id: OntologyMappings-d2079e68-e302-476a
XNIO-1 task-1] o.a.c.Tracing                   INFO        [Trips mappin] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-d2079e68-e302-476a
XNIO-1 task-1] o.a.c.Tracing                   INFO  *<-- [Trips mappin] [from[http://0.0.0.0:8081/rml/vbb/] Exchange[Id: OntologyMappings-d2079e68-e302-476a
XNIO-1 task-2] o.a.c.Tracing                   INFO  *--> [Lowering que] [from[rest://get:/rml/vbb/queries?] Exchange[Id: OntologyMappings-0361a40a-232e-4c33
XNIO-1 task-2] o.a.c.Tracing                   INFO        [Lowering que] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-0361a40a-232e-4c33
XNIO-1 task-2] o.a.c.Tracing                   INFO  *<-- [Lowering que] [from[http://0.0.0.0:8081/rml/vbb/] Exchange[Id: OntologyMappings-0361a40a-232e-4c33
XNIO-1 task-3] o.a.c.Tracing                   INFO  *--> [Fares mappin] [from[rest://get:/rml/vbb/fares?co] Exchange[Id: OntologyMappings-679e224b-fcc5-4d9d
XNIO-1 task-3] o.a.c.Tracing                   INFO        [Fares mappin] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-679e224b-fcc5-4d9d
XNIO-1 task-3] o.a.c.Tracing                   INFO  *<-- [Fares mappin] [from[http://0.0.0.0:8081/rml/vbb/] Exchange[Id: OntologyMappings-679e224b-fcc5-4d9d
XNIO-1 task-4] o.a.c.Tracing                   INFO        [Lowering que] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-6fe63ccd-b2fc-435f
XNIO-1 task-5] o.a.c.Tracing                   INFO        [Trips mappin] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-19c6a7a5-6939-4d4c
XNIO-1 task-6] o.a.c.Tracing                   INFO        [Fares mappin] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-d5c3e741-24c2-41e1
XNIO-1 task-7] o.a.c.Tracing                   INFO  *--> [Functions ma] [from[rest://get:/rml/vbb/function] Exchange[Id: OntologyMappings-868d12d6-005f-4784
XNIO-1 task-7] o.a.c.Tracing                   INFO        [Functions ma] [bean:resourceGetter?method=getRes] Exchange[Id: OntologyMappings-868d12d6-005f-4784
XNIO-1 task-7] o.a.c.Tracing                   INFO  *<-- [Functions ma] [from[http://0.0.0.0:8081/rml/vbb/] Exchange[Id: OntologyMappings-868d12d6-005f-4784
```

Figure 31: Ontology Mappings Resolver activity

The next screen-shot shows the trace generated bt the Converter at startup and then the instatiation of the embedded REST client and the calls to the Ontology Mappings Resolver at the time of Lifting and Lowering:



```
                    main] o.a.c.i.e.AbstractCamelContext INFO  Apache Camel 3.2.0 (CamelContext: VBBTrips2TriasConverter) is starting
                    main] o.a.c.i.e.AbstractCamelContext INFO  Tracing is enabled on CamelContext: VBBTrips2TriasConverter
                    main] o.a.c.c.s.SedaEndpoint         INFO  Endpoint seda://lifter is using shared queue: seda://lifter with size: 1000
                    main] o.a.c.c.s.SedaEndpoint         INFO  Endpoint seda://lowerer is using shared queue: seda://lowerer with size: 1000
                    main] o.a.c.c.s.SedaEndpoint         INFO  Endpoint seda://travelexpertcall is using shared queue: seda://travelexpertcall with size: 1000
                    main] e.DefaultStreamCachingStrategy INFO  StreamCaching in use with spool directory: /var/folders/gd/255235t5413g486hgjj2dglr0000gn/T/camel/cam
                    main] o.a.c.i.e.AbstractCamelContext INFO  Route: RML Lifting started and consuming from: seda://lifter
                    main] o.a.c.i.e.AbstractCamelContext INFO  Route: RML Lowering started and consuming from: seda://lowerer
                    main] o.a.c.c.u.DefaultUndertowHost  INFO  Starting Undertow server on http://localhost:8082
                    main] i.undertow                     INFO  starting server: Undertow - 2.0.30.Final
                    main] o.xnio                         INFO  XNIO version 3.3.8.Final
                    main] o.x.nio                        INFO  XNIO NIO Implementation Version 3.3.8.Final
                    main] o.a.c.i.e.AbstractCamelContext INFO  Route: Rest Service started and consuming from: http://localhost:8082/travelexpert/vbb/trips
                    main] o.a.c.i.e.AbstractCamelContext INFO  Route: TravelExpertCall started and consuming from: seda://travelexpertcall
                    main] o.a.c.i.e.AbstractCamelContext INFO  Route: VBB Trips Conversion started and consuming from: direct://startmapping
                    main] o.a.c.i.e.AbstractCamelContext INFO  Total 5 routes, of which 5 are started
                    main] o.a.c.i.e.AbstractCamelContext INFO  Apache Camel 3.2.0 (CamelContext: VBBTrips2TriasConverter) started in 4.540 seconds
                    main] c.c.v.VBBTripsConverterStarter INFO  Started VBBTripsConverterStarter in 7.396 seconds (JVM running for 8.404)
            XNIO-1 task-1] o.a.c.Tracing                 INFO  *--> [Rest Service] [from[rest://get:/travelexpert/vbb] Exchange[Id: VBBTripsConverter-161b58b8-cdd6-
            XNIO-1 task-1] o.a.c.Tracing                 INFO       [Rest Service] [direct:startmapping              ] Exchange[Id: VBBTripsConverter-161b58b8-cdd6-
d #3 - seda://travelexpertcall] o.a.c.Tracing             INFO  *--> [TravelExpert] [from[seda:travelexpertcall]      ] Exchange[Id: VBBTripsConverter-b429fb76-b9d2-
d #3 - seda://travelexpertcall] o.a.c.Tracing             INFO       [TravelExpert] [Processor@0x7f5eae0f             ] Exchange[Id: VBBTripsConverter-b429fb76-b9d2-
d #3 - seda://travelexpertcall] o.a.c.Tracing             INFO       [TravelExpert] [http://fahrinfo.vbb.de           ] Exchange[Id: VBBTripsConverter-b429fb76-b9d2-
d #3 - seda://travelexpertcall] o.a.c.Tracing             INFO  *<-- [TravelExpert] [from[seda://travelexpertcall]    ] Exchange[Id: VBBTripsConverter-b429fb76-b9d2-
ter) thread #1 - seda://lifter] o.a.c.Tracing             INFO  *--> [RML Lifting ] [from[seda:lifter]              ] Exchange[Id: VBBTripsConverter-53db520d-d647-
ter) thread #1 - seda://lifter] o.a.c.Tracing             INFO       [RML Lifting ] [enrich[constant{direct:rmlmapping] Exchange[Id: VBBTripsConverter-53db520d-d647-
ter) thread #1 - seda://lifter] o.a.c.c.h.HttpComponent   INFO  Created ClientConnectionManager org.apache.http.impl.conn.PoolingHttpClientConnectionManager@1a0caf16
ter) thread #1 - seda://lifter] o.a.c.c.h.HttpComponent   INFO  Created ClientConnectionManager org.apache.http.impl.conn.PoolingHttpClientConnectionManager@2d2c5bc8
ter) thread #1 - seda://lifter] o.a.c.c.h.HttpComponent   INFO  Created ClientConnectionManager org.apache.http.impl.conn.PoolingHttpClientConnectionManager@6fbeea49
ter) thread #1 - seda://lifter] .VBBLIftingMappingsInitializer INFO  RML Initialization complete
ter) thread #1 - seda://lifter] c.c.c.v.VBBHacon2TriasLifter   INFO  RML Initialization extract
ter) thread #1 - seda://lifter] o.a.c.Tracing             INFO  *<-- [RML Lifting ] [from[seda://lifter]            ] Exchange[Id: VBBTripsConverter-53db520d-d647-
er) thread #2 - seda://lowerer] o.a.c.Tracing             INFO  *--> [RML Lowering] [from[seda:lowerer]             ] Exchange[Id: VBBTripsConverter-d8bd46ca-1462-
er) thread #2 - seda://lowerer] o.a.c.Tracing             INFO       [RML Lowering] [enrich[constant{direct:loweringpr] Exchange[Id: VBBTripsConverter-d8bd46ca-1462-
er) thread #2 - seda://lowerer] o.a.c.c.h.HttpComponent   INFO  Created ClientConnectionManager org.apache.http.impl.conn.PoolingHttpClientConnectionManager@4af7655f
er) thread #2 - seda://lowerer] .VBBLoweringQueriesInitializer INFO  Lowering Queries Initialization complete
er) thread #2 - seda://lowerer] c.c.c.v.VBBHacon2TriasLowerer  INFO  Lowering queries Initialization extract
er) thread #2 - seda://lowerer] o.a.c.Tracing             INFO  *<-- [RML Lowering] [from[seda://lowerer]           ] Exchange[Id: VBBTripsConverter-d8bd46ca-1462-
            XNIO-1 task-1] o.a.c.Tracing                 INFO  *<-- [Rest Service] [from[http://localhost:8082/travel] Exchange[Id: VBBTripsConverter-161b58b8-cdd6-
```

Figure 32: Converter invoking REST Ontology Mappings Resolver services

In the leftmost column at the bottom the Converter pipeline shown in Figure 25 can be seen in execution progessing from the Travel Expert Call to Lifting and to Lowering, with the latter two getting RML mappings and Lowering Queries from the direct:rmlmapping and direct:loweringpr endpoints respectively. These end point instantiate ClientConnectionManager to invoke the REST Ontology Mappings Resolver services.

The following is, finally, a screen-shot of the JConsole management application connected to the Ontology Mappings Resolver displaying essential statistics on the health status and performance of the service (processing times expressed in milliseconds)

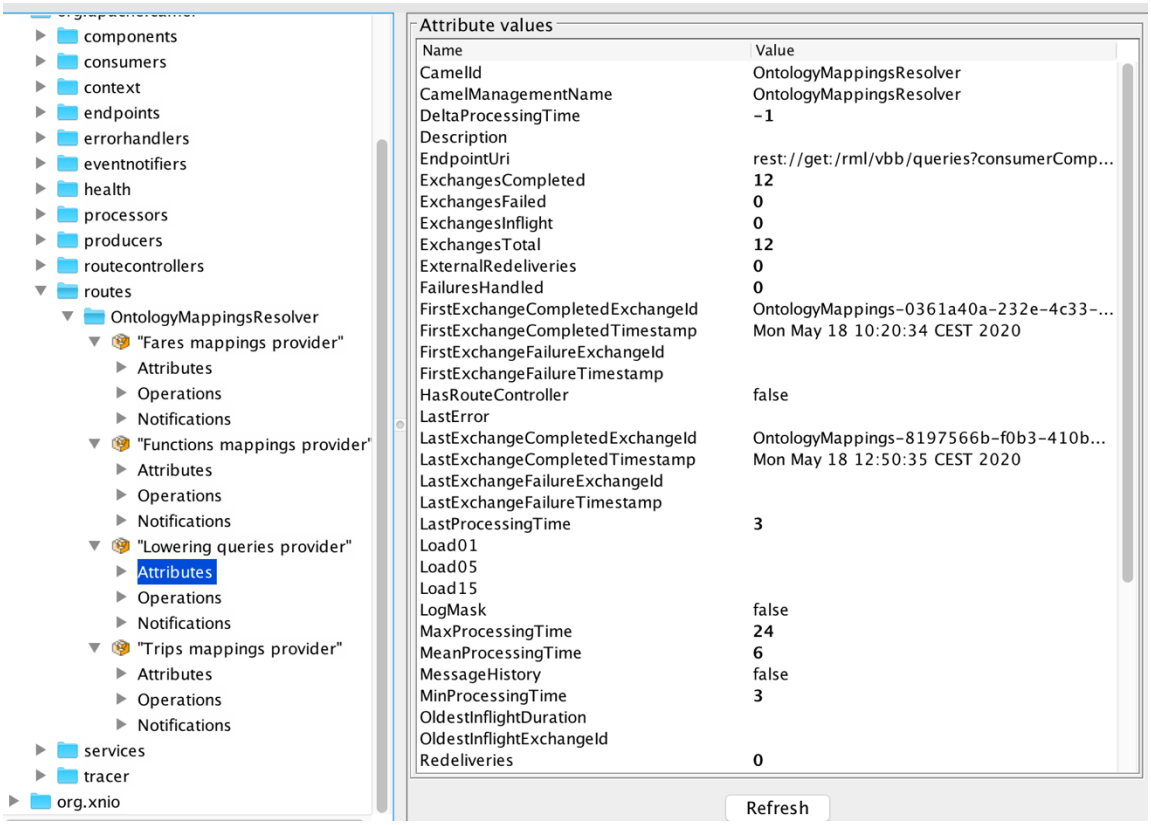


Figure 33: JConsole for Ontology Mappings Resolver

3. PERFORMANCE AND SCALABILITY EVALUATION

In the previous section, we functionally validated components of the IF against scenarios and requirements elicited for C-Rel. This section describes the preliminary performance and scalability tests performed to evaluate artifacts developed for C-Rel also considering scenarios and KPIs identified in the Deliverable 3.2 [1] and the testing infrastructure described in Deliverable 3.3 [2]. This activity has a double aim, on one hand, it tests the current implementation on a given set of metrics, on the other hand, it aims at identifying bottlenecks and issues to gather new requirements for the final release.

In the context of IF performance and scalability evaluation, it is of foremost importance to analyze components that are expected to be stressed in a production-ready IF deployment. Therefore, considering C-Rel artifacts, the evaluation performed for C-Rel mainly focuses on two critical aspects for the IF and the related components, i.e., querying and converting heterogeneous data.

3.1 QUERYING HETEROGENEOUS DATA

This section is focused on the Distributed SPARQL endpoint, one IF core components whose function is to answer SPARQL queries on various heterogeneous and autonomous data sources in the Data Layer. However, in the transport domain, data sources can be very broad in terms of the number of sources, data volume, heterogeneity and the ways in which the data can be accessed. Thus, collecting and integrating all this large volume of data is a costly task for any organization including transport authorities. To facilitate the task of data integration, a Distributed SPARQL endpoint provides a conceptual and integrated view (knowledge graph) from the various data sources, hiding details of each source and allowing queries on a common representation of all data sources. Under the Ontology-based data integration (OBDI) approach, data integration is achieved through the use of mappings between the knowledge graph and the data sources. Although in the state of the art, there is the approach for knowledge graph materialization [3], in the last decades the need to virtualize the knowledge graph has arisen because data can constantly change or even integrate new data sources to the OBDI system. If the materialization is used, the knowledge graph must be reconstructed for each update of the data or incorporation of new data sources. Since the size of the knowledge graph can be very large in the transport domain, this materialization would be very costly given the reconstruction of the knowledge graph [4]. On the other hand, updating the data does not affect the virtualization approach since the knowledge network is virtual and only retrieves the data when it is needed, that is, when a query is executed on the OBDI system and therefore the query extracts the updated data. Like materialization, if a new data source is incorporated, mappings must be defined. Given the advantages of the virtual approach, we have designed and applied a testbed to evaluate the performance and scalability of the Distributed SPARQL endpoint and also to compare it with other engines of the state-of-the-art that virtualize the knowledge graph.

In the design of our testbed we have decided to work with data sources that are not particularly attached to the S2R domain, but share many of its characteristics, especially with the objective that the testbed provides a neutral view over the whole transport domain (so that other researchers can also make use of it more easily to take decisions on the systems to use for a particular task) while providing useful insights for the core technology components that may be part of the implementation of the aforementioned components of the S2R IF.

Therefore, we have decided to use data about public transport in the city of Madrid. More specifically, the web portal of Madrid Regional Transportation Consortium has published information about public transport of Madrid in order to users and not-for-profit enterprises can find and reuse these data. With this in mind, a benchmark for virtual knowledge graph access in the transport domain following the General Transit Feed Specification (GTFS) has been defined by us. The result of this work is also published at [5].

As in any other similar testbed, we will describe the datasets and mappings used in the testbed and the queries used to analyse the system behaviour.

3.1.1 Dataset

The datasets are based on the GTFS (General Transit Feed Specification) [6] and they are composed by a set of files in 4 formats: CSV, JSON, SQL, and XML. The data were extracted from the metro of the Madrid city. Basically, the GTFS is a standard developed by Google and it specifies a common format for public transportation schedules and associated geographic information. GTFS is composed of a set of files where each one represents aspects related to transit information.

On the other hand, 5 different instances in CSV format were generated from the dataset of the Madrid metro based on GTFS (original dataset) in order to measure and analyze scalability. The original dataset instance was scaled to 5, 10, 50, 100 and 500 times its initial size using VIG [7] as a tool to generate datasets with different scale values while taking account the domain information of ontologies and mappings. Then, we have applied open tools to transform the output of VIG, CSV files, to different formats (JSON, SQL, and XML). All the generated data is publicly available[10]. The rest of the variables of the dimension are constant: number of sources are always 10, partitioning type is vertical and the data distribution is maintained due the use of VIG in the generation of the different sizes.

3.1.2 Queries

A set of 18 queries were expressed in SPARQL which is presented in Table 6 in terms of the number triple patterns, the number of data sources selected, the use of OPTIONAL clause or aggregation functions, other features, equality (equal to) or range (relational) conditions in the FILTER clause, and the number of constants in a query. Star-shaped group is a group of triple patterns that are “joined" over the same subject or object variable. Particularly, the number of triple patterns varies from 3 to 15, the number of sources is between 1 and 5, the other features are DISTINCT, NOT EXIST, GROUP BY, ORDER BY and UNION, and the number of constants is between 0 and 5.

[10] https://github.com/oeg-upm/gtfs-bench/tree/master/data

Table 6: GTFS-Benchmark Queries. Retrieved from [5]

Query	Description	#Triple Patterns	#Sources	OPTIONAL	Aggregation	Other features	FILTER		#Star-shaped groups	
							equal to	relational	w/o constants	w/ constants
1	All agencies	4	1						1	0
2	All stops between two locations	5	1	✓				✓	0	1
3	Accessibility information of a specific stop	5	1	✓			✓		0	1
4	All agencies and their routes	9	2	✓					2	0
5	Services that have been added on a specific date	5	2					✓	1	1
6	Number of routes covered by a specific agency	3	2		Y		✓		0	2
7	All wheelchair-accessible stops in a specific route	15	4	✓		DISTINCT	✓		1	3
8	All routes that pass through a specific stop	14	5	✓					5	0
9	Given a specific stop, get the arrival times of vehicles going in a particular direction	7	2	✓				Y	1	1
10	Number of services offered by a specific route in a particular period	4	2		✓	DISTINCT		Y	1	1
11	Trips of a specific route that are available on a certain date	12	3			NOT EXISTS		Y	3	2
12	Number of stops that are wheelchair-accessible grouped by route	10	4		✓	GROUP BY			3	1
13	All the accesses of a specific station	6	1	✓					0	1
14	For a specific line, all the stops from its origin, in a specific direction and service	8	3	✓		ORDER BY			3	0
15	For all properties, triples that contain a specific word in the object placeholder	3	1				✓		0	1
16	For all routes, all the calendar changes done in December	8	3				✓		0	1
17	For all services, duration of a specific route between two particular stops	9	3						3	0
18	All routes that have trips on Sunday	8	5			UNION			4	1

3.1.3 Mappings

A set of mappings in different languages (RML [8], R2RML [9], xR2RML [10], and Ontop [11] OBDA mappings) was used to map from the GTFS-based data sources to the Linked GTFS ontology [11]. The Linked GTFS vocabulary can be seen as a representation of the GTFS specification as an ontology.

Most of the classes of the Linked GTFS ontology were implemented. Nevertheless, the classes gtfs:FareClass, gtfs:FareRule and gtfs:RouteType are not considered because Madrid GTFS does not include information on fares and the data covers only the Metro system.

[11] https://github.com/OpenTransport/linked-gtfs

Finally, a total of 9 mapping files[12] were generated from the Linked GTFS to MYSQL, CSV, XML and JSON using the languages RML, R2RML, xR2ML and Ontop OBDA model.

Based on the datasets, mappings and queries described in Sections 3.1.1-3.1.3, the behaviour of five open-source query engines were experimentally evaluated. These engines are:

- Ontario v.0.3[13]: It is a federated query engine over heterogenous data sources that supports OBDI (Ontology Based Data Integration) and whose data sources are described by RDF Molecule Templates. An RDF Molecule Template corresponds to an abstract description of properties associated with the same type of RDF Molecules and an RDF Molecule is a set of RDF triples that share the same subject. Using a set of heuristics, Ontario is guided to select the best data sources on which will be generated sub queries to be translated and subsequently executed in the underlying query languages of the selected data sources. Additionally, the data can be stored in several formats: RDF, MySQL, CSV, TSV, JSON, XML, MongoDB and Neo4j.

- Ontop v3.0.0[14]: It is an OBDA system where datalog rules represent mappings between a relational data source and a virtual (or materialised) knowledge graph. It only considers the SQL format and it is able to generate more efficient SQL queries thanks to several optimization techniques implemented in it.

- Morph-RDB v3.12.5[15]: It is an OBDA system based on the R2RML standard. It implements query optimization techniques on data sources stored as SQL and CSV files.

- Morph-CSV v1.0.0[16]: It is an OBDA system that exploits the information of CSVW annotations and RML+FnO mappings to create an enriched RDB representation of the CSV files together with the corresponding R2RML mappings, enabling the use of existing query translation (SPARQL-to-SQL) techniques implemented in R2RML-compliant OBDA engines.

- Morph-xR2RML-1.1-RC2[17]: It is an OBDA system based on xR2RML for non-NoSQL databases such as MongoDB

All experiments were performed using Docker containers to ensure reproducibility. For each query engine, a docker image was created considering the recommended setting provided in the corresponding online repository. For each SQL dataset size, two docker images were created, one as an instance of the MySQL Database Server v5.5 and another as an instance of the MySQL Community Server v8.0. Similarly, for each MongoDB dataset size, a docker image of an instance of the MongoDB Community Server v3.4 was created. and the dataset is loaded. The datasets were loaded to their correspondent database servers. With respect to the raw data (CSV, XML and JSON),

[12] https://github.com/oeg-upm/gtfs-bench/tree/master/mappings

[13] https://github.com/SDM-TIB/Ontario

[14] https://github.com/ontop/ontop

[15] https://github.com/oeg-upm/morph-rdb

[16] https://github.com/oeg-upm/morph-csv

[17] https://github.com/frmichel/morph-xr2rml

they were loaded into the machine and are accessible to all the engines. In the case of Morph-RDB, it was used together with the docker images containing the instances of the MySQL Community Server v5.5, according to the corresponding documentation. As for Morph-xR2RML, it was used with the docker images containing the instances of MongoDB server version v3.4.

Additionally, the 18 SPARQL queries were evaluated in warm and in cold mode in order to analyze how the cache mechanism may affect the performance of the engines. Each query is run five times. In warm mode, each query was evaluated discarding its first run and then it was run again 5 times to compute the average query execution time. In cold mode, the database server was restarted after each run to clean all the caches. It is noteworthy that the docker images of Ontario and Ontop contain instances of MySQL server v8.0. For these instances, the queries were executed in cold cache since the use of cache is not supported anymore in MySQL v8.0

Finally, experiments were executed on a machine with the following characteristics: 2GHz CPU with 15 cores, 32 RAM, 200 GB HDD with Ubuntu 18.04 as its operating system.

3.1.4 Performance Test Results and Analysis

Ontology Based Data Access (and Integration) systems aim at presenting a conceptual common view from a set of heterogenous and autonomous data sources. In this context, IF data layer relays on a data collection from many sources which will be described by a reference ontology of the transportation domain in order to allow to discover, navigate and query heterogeneous data sources. In this regard, the proposed testbed allows to evaluate the performance and scalability of this type of solutions that can be integrated into S2R IF. Thus, the testbed has been executed to empirically evaluate five tools (Ontario, Ontop, Morph-RDB, Morph-CSV and Morph-xR2RML) that provide a unified access to heterogeneous data sources (CSV, JSON, SQL and XML).

Table 7 shows the average execution time obtained for 18 SPARQL queries in warm and in cold mode considering different formats on the dataset of the Madrid metro based on GTFS (original dataset) and the five Ontology based tools; query timeout was 3600s (1 hour). When a tool reports an error (e.g. a SPARQL query parsing error or memory overhead), it is reported as E in the Table 7. If a tool produces a different number of results in comparison to the baseline, a W is shown in Table 7. TO means timeout has occurred.

Table 7: Average execution time (sec) of testbed queries with original size datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	5.85	2.07	E	1.82	W	1.86	1.97	E	26.02	1.80	E	1.81	2.06	W	1.89	E	2.11	E
	Cold	Ontario	18.02	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
		Morph-RDB	7.14	2.65	E	2.42	W	2.36	2.43	E	28.65	2.38	E	2.41	2.69	W	2.58	E	2.68	E
		Ontop	8.37	5.04	5.18	E	W	E	W	E	16.56	E	E	E	5.06	W	5.10	W	5.00	W
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	28.67	W	W	6.52	W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	28.17	W	W	6.96	W
GTFS-CSV	Cold	Morph-RDB	6.94	3.04	E	2.78	E	2.78	TO	E	TO	2.97	E	6.23	3.97	E	E	E	3.14	W
		Morph-CSV	15.11	10.88	E	10.72	E	9.95	10.84	E	40.90	10.70	E	11.60	11.82	E	E	E	11.48	W
		Ontario	W	E	17.34	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	18.04	E	17.14	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	17.14	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	17.14	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

We can observe in Table 7 that most of the engines are able to answer the majority of the queries when the queries are executed on CSV and SQL datasets. Also, more time is required to execute queries over CSV datasets than SQL ones because engines load the CSV dataset in a SQL database server before answering the queries. In the case of JSON, Mongo and XML datasets, the engines are only capable of answering two queries at most. Finally, there is not much difference between warm and cold cache possibly due to size of the datasets.

3.1.5 Scalability Test Results and Analysis

Table 8-

Table 12 show the average execution time obtained for 18 SPARQL queries in warm and in cold mode considering different formats on datasets scaled to 5-500 times of the original dataset size and the five Ontology based tools. Also, query timeout was 3600s (1 hour). When a tool reports an error, it is indicated as E. If a tool produces a different number of results in comparison to the baseline, a W is shown in the table. TO means timeout has occurred. For these cases that the query reports 0 results in the execution but without error, the total execution time is not indicated.

The engines hold their behaviour in the other scale factors up to 100. In the scale factor 500, only those engines that use SQL datasets are able to answer queries. It is noteworthy that not all of OBDA/OBDI engines were able to answer the queries because they do not support some SPARQL operators such as UNION, ORDER BY and NOT EXISTS. The best behaviour was obtained by those engines that support SPARQL-to-SQL query translation possibly because this type of technique has been widely studied in the state of art. The rest of the engines that translate queries over raw data (e.g. CSV, XML) have produced errors during the query execution process.

Table 8: Average execution time (sec) of testbed queries with size 5 datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	12.65	2.47	E	1.89	2.06	1.78	1.93	E	W	1.74	E	1.88	2.14	4.58	2.88	E	2.61	E
	Cold	Ontario	117.00	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
		Morph-RDB	15.14	3.24	E	2.40	2.71	2.34	2.62	E	W	2.41	E	2.70	2.82	5.59	3.89	E	3.39	E
		Ontop	13.87	5.40	5.31	E	W	E	W	E	W	E	E	E	5.24	6.61	W	W	5.37	4.77
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
GTFS-CSV	Cold	Morph-RDB	14.42	4.38	E	3.81	E	3.64	TO	E	TO	6.57	E	TO	12.45	E	E	E	9.25	W
		Morph-CSV	43.41	W	E	33.51	E	34.44	W	E	TO	33.86	E	36.08	34.90	E	E	E	35.26	W
		Ontario	W	E	18.34	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	W	E	15.66	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	15.66	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	15.66	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

Table 9: Average execution time (sec) of testbed queries with size 10 datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	23.78	2.88	E	1.93	W	1.75	1.97	E	W	1.85	E	1.94	2.46	6.61	3.46	E	3.07	E
	Cold	Ontario	415.60	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
		Morph-RDB	27.25	3.72	E	2.54	W	2.36	2.55	E	W	2.38	E	2.50	3.22	8.16	4.48	E	3.77	E
		Ontop	24.05	5.56	5.57	E	W	E	W	E	W	E	E	E	5.29	7.58	W	W	5.62	W
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W		W	W		W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W		W	W		W
GTFS-CSV	Cold	Morph-RDB	25.90	6.06	E	5.20	E	4.89	TO	E	TO	16.06	E	TO	38.15	E	E	E	38.90	W
		Morph-CSV	97.00	W	E	69.39	E	68.78	W	E	TO	69.28	E	71.01	68.79	E	E	E	72.29	W
		Ontario	W	E	19.51	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	W	E	17.21	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	17.21	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	17.21	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

Table 10: Average execution time (sec) of testbed queries with size 50 datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	108.42	4.91	E	2.08	W	1.75	1.97	E	W	1.89	E	2.29	3.69	22.55	8.27	E	5.56	E
	Cold	Ontario	TO	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	W	E
		Morph-RDB	121.31	6.01	E	2.68	W	2.31	2.63	E	W	2.59	E	2.91	4.54	27.02	10.00	E	6.89	E
		Ontop	119.89	6.92	6.61	E	W	E	W	E	W	E	E	E	6.05	15.69	W	W	7.31	W
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W
GTFS-CSV	Cold	Morph-RDB	128.40	22.17	E	19.85	E	19.60	TO	E	TO	351.23	E	TO	1039.29	E	E	E	TO	W
		Morph-CSV	575.15	449.54	E	442.60	E	436.06	W	E	TO	444.84	E	443.12	447.74	E	E	E	443.47	W
		Ontario	W	E	35.16	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	W	E	23.74	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	23.74	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	23.74	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

Table 11: Average execution time (sec) of testbed queries with size 100 datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	221.11	7.48	E	2.30	W	1.75	1.96	E	W	1.99	E	2.65	4.68	42.44	15.51	E	8.54	E
	Cold	Ontario	TO	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
		Morph-RDB	245.98	8.83	E	3.05	W	2.33	2.52	E	W	2.63	E	3.38	5.76	50.99	19.45	E	10.38	E
		Ontop	1477.38	8.87	8.25	E	W	E	W	E	W	E	E	E	6.80	27.18	W	W	9.20	4.58
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W
GTFS-CSV	Cold	Morph-RDB	E	43.59	E	38.52	E	38.43	TO	E	TO	1582.52	E	TO	TO	E	E	E	TO	W
		Morph-CSV	1254.19	W	E	958.43	E	933.69	W	E	TO	957.95	E	951.53	952.93	E	E	E	947.82	W
		Ontario	W	E	85.59	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	W	E	33.56	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	33.56	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	33.56	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

Table 12: Average execution time (sec) of testbed queries with size 500 datasets. Retrieved from [5]

Dataset	Cache	Engine	Queries																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
GTFS-SQL	Warm	Morph-RDB	TO	29.85	E	3.39	W	1.81	1.96	E	W	3.19	E	6.34	13.60	220.35	93.72	E	33.64	E
	Cold	Ontario	TO	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
		Morph-RDB	TO	32.71	E	3.92	W	2.09	2.30	E	W	3.62	E	6.95	14.69	218.00	99.00	E	35.77	E
		Ontop	W	20.93	17.17	E	W	E	W	E	W	E	E	E	10.82	114.59	W	W	23.95	W
GTFS-MongoDB	Warm	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W	W	TO	W
	Cold	Morph-xR2RML	W	W	W	W	W	W	W	W	W	W	W	W	W	TO	W	W	TO	W
GTFS-CSV	Cold	Morph-RDB	E	TO	E	TO	E	TO	TO	E	TO	TO	E	TO	TO	E	E	E	TO	W
		Morph-CSV	TO	W	E	TO	E	TO	W	E	TO	TO	E	TO	TO	E	E	E	TO	W
		Ontario	W	E	E	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-XML	Cold	Ontario	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
GTFS-JSON	Cold	Ontario	W	E	E	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-B	Cold	Ontario	W	E	E	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-W	Cold	Ontario	W	E	E	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E
GTFS-R	Cold	Ontario	W	E	TO	E	E	E	E	W	E	E	E	E	E	W	E	E	E	E

In summary, we have applied a benchmark for the access of virtualized Knowledge Graph in the transport domain. The benchmark receives as input a set of queries, mappings and datasets in different formats in order to test the capabilities and performance of the state of the art engines.
We have also followed the steps of the benchmark described in Section 4.2 of the D3.3, to illustrate the workflow for access of the virtualised knowledge graph and have established a set of metrics that will measure the performance and scalability of one of the IF core components: The Distributed SPARQL endpoint.
We have applied the benchmark in order to compare the behaviour of the Distributed SPARQL endpoint with respect to other engines of the state-of-the-art and based on the results obtained we can conclude that the state-of-the-art engines are not yet sufficiently mature and there are still relevant aspects to be addressed:

- With respect to heterogeneity, the engines of the state-of-the-art are not able to respond satisfactorily to queries when the data is JSON and XML because they produce incomplete or empty responses in many cases. The best behaviour was for relational database case, followed by data in CSV format.
- Another important aspect is that the engines of the state-of-the-art do not support SPARQL 1.1 completely. For example, SPARQL queries with "FILTER NOT EXISTS" can not be resolved. As these engines must translate from SPARQL queries to underlying engine queries, many of them still have to improve their translation process.
- In some cases, the query translation is performed naively without optimizations by some engines of the state-of-the-art and therefore there is a need to include optimizations in the query translation process as part of the development of these tools.
- Due to the lack of maturity in the virtualization approach, there is no tool that covers all these optimization needs in query translation, and full support of SPARQL 1.1 and heterogeneous data.

Finally, the results of our benchmark also show the necessity for improvements of state-of-the-art engines in terms of query completeness and efficiency.

3.2 CONVERTING HETEROGENEOUS DATA

The evaluation of converter performance and scalability considers the two main scenarios described in deliverable 3.2:

- *Batch Data Conversion*: this scenario considers the case where a batch dataset should be converted. Usually, this scenario doesn't have particular constraints on the conversion time but requires scalability with respect to the size of the dataset that can be in the order of hundreds of megabytes.
- *Runtime Data/Message Conversion (Service Mediation)*: this scenario considers the case where a message or a small amount of data (in the order of bytes/megabytes) should be converted to guarantee communication between two different systems. Usually, this scenario involves small size datasets but requires conversion time to be as short as possible to introduce low overhead in the communication.

As described in deliverable 5.2 [12] Chimera, the proposed implementation for the IF Converter component, adopts a modular approach to build flexible pipelines for conversions based on Semantic Web technologies. In particular, the C-Rel release implements two alternatives exploiting a materialization approach for lifting (RML-Mapper and ST4RT annotations) and two alternatives for lowering (Template-based and ST4RT annotations).

As described in deliverable 3.3 [2], we planned testing activities for the converter leveraging different datasets and related mappings in the *batch data* scenario and in the *runtime data/message* scenario. Considering the already implemented blocks in Chimera we decided to carry out a first subset of the proposed testing activities to validate the C-Rel implementation and its performance and scalability requirements:

1. *Batch Data Conversion*: Roundtrip conversion GTFS-Linked GTFS-GTFS with Chimera, using materialization lifting via RML-Mapper and Apache Velocity template lowering over materialized RDF.
2. *Runtime Data/Message Conversion (Service Mediation)*: FSM/918 conversion developed in the ST4RT project and ported in Chimera, using materialization through annotations via ST4RT lifting block, and lowering through annotations via ST4RT lowering block.

Considering the first testing activity, in Section 3.2.1, we describe the datasets used in terms of their sizes to provide an overview of the scale of the overall testing activity. In Section 3.2.2, we will report tests for the described scenario discussing how different sizes and formats affect the Chimera implementation. In the implemented pipeline, GTFS feed is read from the filesystem, the feed is unzipped, lifting RML mappings are executed to materialize the RDF graph, a set of templates (one for each GTFS file) is executed in parallel to populate the resulting files querying the RDF graph, results of the lowering phase are written to the filesystem.

Additionally, we will propose two pairs of tests and results executed to better understand the bottlenecks of the solution. In Section 3.2.3, we will compare the Chimera lifting phase with RML-Mapper in isolation and with SDM-RDFizer[18], an alternative mapper satisfying the RML specification.

[18] https://github.com/SDM-TIB/SDM-RDFizer

This will allow us to check the overhead introduced by Chimera and to discuss and compare the adopted library with different implementations. In Section 3.2.4, we will compare the executed conversion using a different set of RML mappings optimized to avoid *join* conditions and, thus, reducing the computational load required for the execution. Last but not least, in Section 3.2.5 we discuss in details bottlenecks related to materialization from XML files.

Considering the second testing activity, in Section 3.2.6, we discuss results obtained running the same tests used in the ST4RT project for the conversion FSM/918. This activity allows us to validate the porting of the annotation-based solution in Chimera and to compare performances obtained.

All tests were run using Docker Containers on a machine running CentOS Linux 7, with Intel® Xeon 8-core CPU and 64 GB Memory. Memory constraint is set to 24GB using the Docker *–memory-limit* option on containers in execution, no limits are set on CPU usage. A timeout of 24 hours is set for each conversion executed. Each test has been run 5 times.

Data gathered are collected using UNIX-provided statistics on running processes and through the collection of timestamps obtained instrumenting the Chimera pipeline. Moreover, we monitored containers in execution using the Telegraf[19] agent to gather data from the Docker Daemon, InfluxDB[20] to store time-series and Grafana[21] to plot them.

3.2.1 The GTFS-Madrid-Bench CSV dataset

The datasets used in the testing activities performed for the batch data conversion scenario are the ones of the GTFS-Madrid-Bench described in Section 3.1.1. In particular, we focused on the three most common format, i.e., CSV, JSON and XML. For each format, we considered different size datasets scaling the original dataset (scale 1) with scales 5, 10, 50, 100, 500. In Figure 34, for each file in each dataset, we report the number of lines and the size of the unzipped file. The last row shows the aggregates for each row.

It is important to point out that the generated datasets are built for performance testing purposes, however, a typical GTFS feed is in the order of tens of megabytes. The GTFS feed of a transport operator rarely overcomes the 100 MB unzipped.

[19] https://www.influxdata.com/time-series-platform/telegraf/

[20] https://www.influxdata.com/products/influxdb-overview/

[21] https://grafana.com/

The JSON and XML corresponding versions of each CSV dataset contain the same information but have higher sizes since they are encoded using a more verbose format. For example *1-json* size is 10.7 MB, *1-xml* size is 15.1 MB.

	1-csv.zip	5-csv.zip	10-csv.zip	50-csv.zip	100-csv.zip	500-csv.zip
AGENCY.csv	1 (0.0MB)	5 (0.0MB)	10 (0.0MB)	50 (0.01MB)	100 (0.01MB)	500 (0.05MB)
CALENDAR.csv	5 (0.0MB)	25 (0.0MB)	50 (0.0MB)	250 (0.01MB)	500 (0.04MB)	2500 (0.14MB)
CALENDAR_DATES.csv	70 (0.0MB)	350 (0.01MB)	700 (0.03MB)	3500 (0.12MB)	7000 (0.32MB)	35000 (1.19MB)
FEED_INFO.csv	1 (0.0MB)	5 (0.0MB)	10 (0.0MB)	50 (0.0MB)	100 (0.01MB)	500 (0.04MB)
FREQUENCIES.csv	855 (0.05MB)	4275 (0.3MB)	8550 (0.58MB)	42750 (2.96MB)	85500 (5.86MB)	427500 (29.89MB)
ROUTES.csv	13 (0.0MB)	65 (0.01MB)	130 (0.02MB)	650 (0.13MB)	1300 (0.22MB)	6500 (1.28MB)
SHAPES.csv	58540 (4.55MB)	176830 (8.26MB)	353660 (17.35MB)	1768300 (84.4MB)	3536600 (184.07MB)	17683000 (861.72MB)
STOP_TIMES.csv	2364 (0.16MB)	11820 (0.82MB)	23640 (3.16MB)	118200 (8.29MB)	236400 (31.83MB)	1182000 (84.0MB)
STOPS.csv	1262 (0.13MB)	6310 (0.93MB)	12620 (2.29MB)	63100 (9.33MB)	126200 (23.19MB)	631000 (93.97MB)
TRIPS.csv	130 (0.01MB)	650 (0.09MB)	1300 (0.2MB)	6500 (0.85MB)	13000 (1.98MB)	65000 (8.52MB)
ALL	63241 (4.9MB)	200335 (10.42MB)	400670 (23.64MB)	2003350 (106.1MB)	4006700 (247.53MB)	20033500 (1080.8MB)

Figure 34: Number of elements and sizes of each file in input datasets 1,5,10,50,100,500 CSV.

3.2.2 Materialization Approach considering Batch Data in Chimera

This section reports results obtained testing roundtrip conversion GTFS-Linked GTFS-GTFS with Chimera. First of all, GTFS to Linked GTFS RML mappings are used to implement materialization lifting via RML-Mapper. Then, a set of Apache Velocity templates are used to query the materialized RDF represented using the LinkedGTFS ontology and to lower data reassembling the original GTFS files.

In Table 13, the complete results of conversion execution for the different sizes and different formats are provided. Data reported are measured in seconds and they represent the average execution time on the 5 attempts performed for each conversion. TO stands for timeout and it means that the conversion was stopped since it reached the 24-hours timeout.

As it may be noticed, for XML even scale 1 dataset failed to complete the conversion within the set timeout. For this reason, we will focus mainly on CSV and JSON in this section and we will investigate more in detail the XML problem in Section 3.2.5.

Table 13: Complete results of conversion execution with Chimera (RML-Mapper lifting and Template-based lowering)

	1	5	10	50	100	500
CSV	22.77s	164.95 s	544.41s	11624.45s	TO	TO
JSON	50.41s	659.11s	2471.29s	66003.36s	TO	TO
XML	TO	TO	TO	TO	TO	TO

To provide comparable numbers, CSV times go from 23 seconds for 1-scale dataset to 3 hours for the 50-scale dataset, JSON times go from 50 seconds for 1-scale dataset to 18 hours for the 50-scale dataset. We can assert that up to 50-scale the conversion is executed in a reasonable amount of time for the *Batch conversion* scenario for CSV and JSON datasets. In the following sections, we will discuss in detail bottlenecks and possible improvements.

Before presenting more detailed visualizations on conversions executed, in Figure 35 we show the numbers of triples materialized for each dataset ending the conversion within the timeout. As expected, the number of triples generated from CSV and JSON datasets with the same scale are the same. Moreover, the growth in the number of triples is proportional to scale growth.

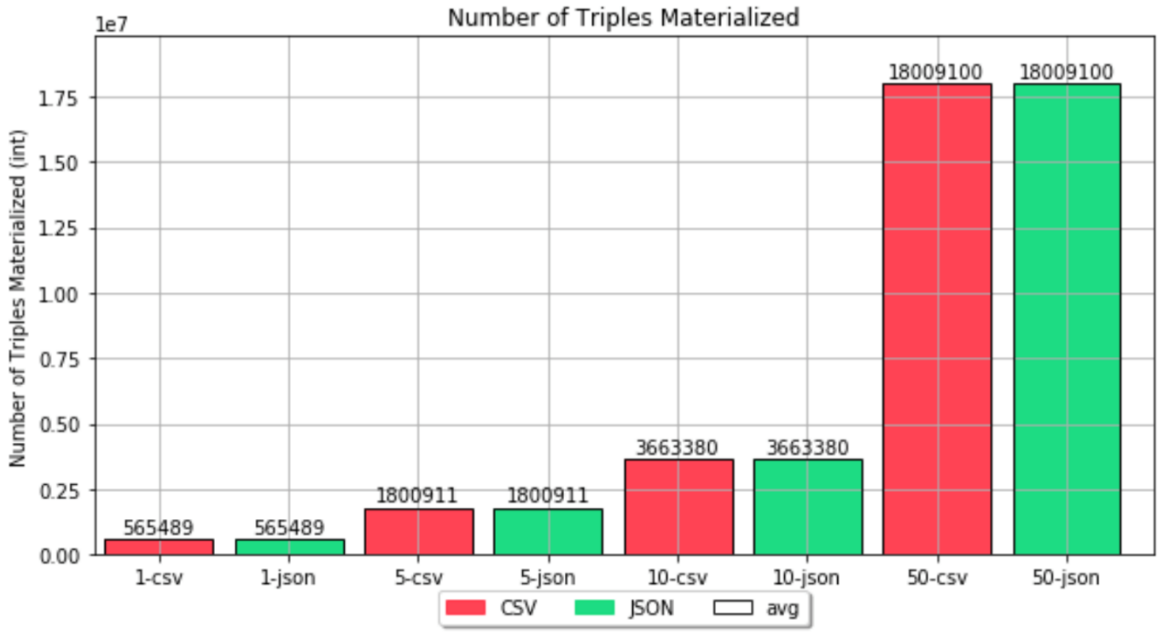


Figure 35: Number of triples materialized with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

In Figure 36, we reported a graphical comparison of execution times for CSV and JSON datasets with scale 1,5,10. In Figure 37 we considered also 50-scale reporting it in a separate figure since it abundantly exceeds other sizes considered.

In these graphs, each black-outlined rectangle represents the average value obtained considering the five executions for the case reported on the x-axis. Each black-outlined rectangle is composed of five adjacent bars representing the exact value obtained in the n-th execution. The type of graph used in these two figures will be used to report multiple tests results changing the x- and y-axis and the color legend reported below.

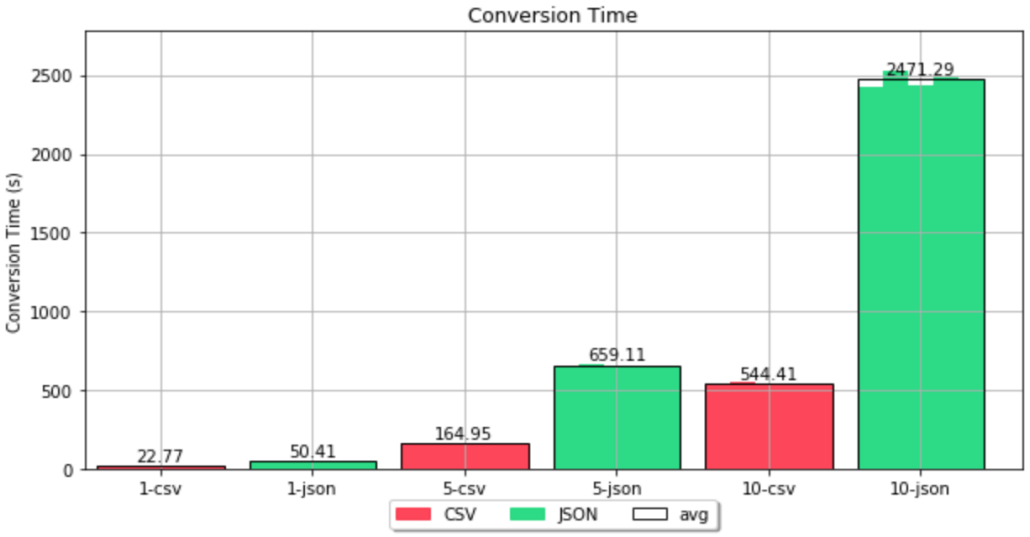


Figure 36: Conversion Time Chimera (RML-Mapper and Template Lowering) with 1,5,10-scale CSV and 1,5,10-scale JSON datasets.

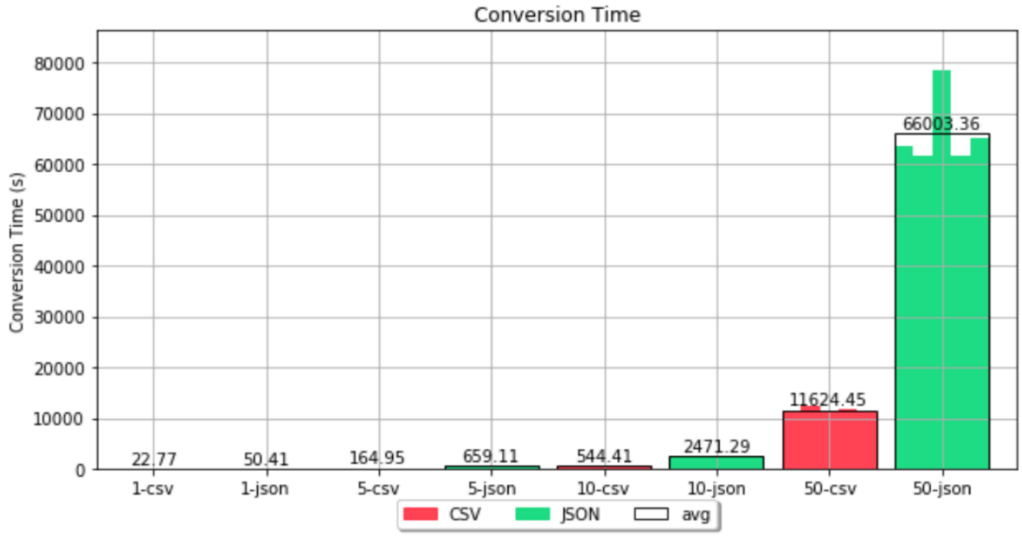


Figure 37: Conversion Time Chimera (RML-Mapper and Template Lowering) with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

A first important aspect that can be noticed is that the growth is not linear with respect to the dataset's scale growth, i.e. execution time of 5-csv is more than five times the one of 1-csv. A second aspect is related to the fact that CSV conversion time outperforms JSON in the 1-scale dataset, and the CSV growth-rate considering different scale datasets is lower with respect to the JSON one.

To investigate which can be the root causes behind these observations, and to visualize how lifting and lowering affect the overall execution time, in Figure 38, we reported the following comparison: for each dataset, considering average values on the 5 executions, we compare lifting, lowering and total conversion times. The obtained graph showcases how the lifting phase is responsible almost entirely of the total execution time. This result suggested us to focus on the lifting procedure to identify possible performance bottlenecks.

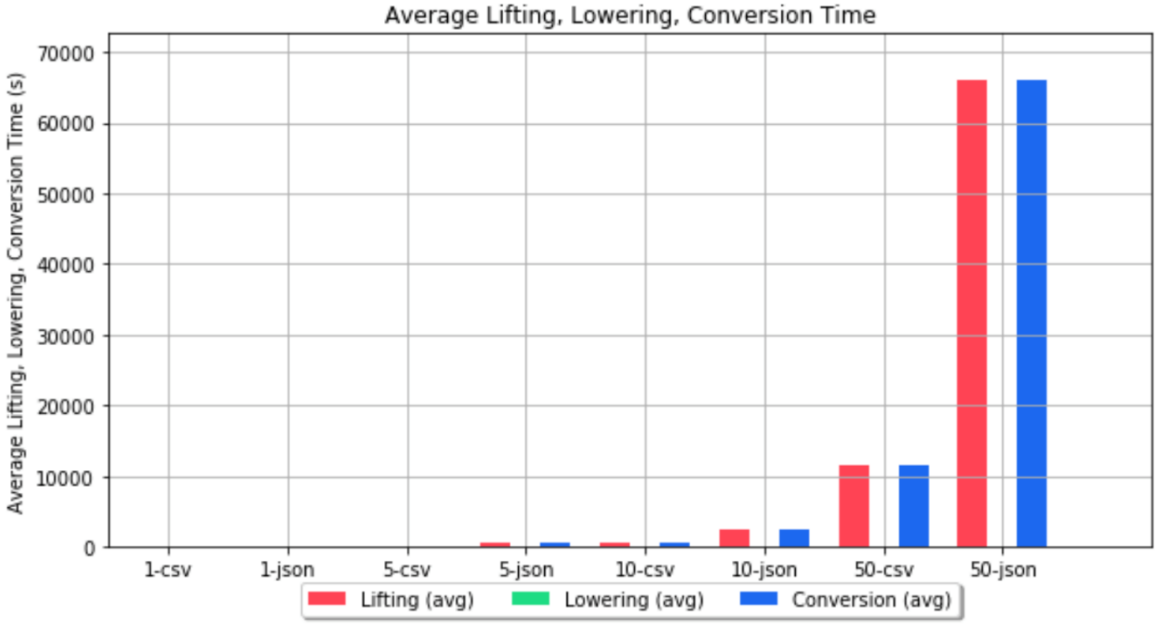


Figure 38: Lifting, Lowering, Conversion Time Chimera (RML-Mapper and Template Lowering) with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

Analyzing the implementation of the RML-Mapper library used for the related lifting block in Chimera, two main bottlenecks may be identified considering observations made above:

1. ***Joins in mappings***. The not-linear growth with respect to the scale growth of the dataset is mainly due to the usage of *join*s condition in the RML mappings to build triples among subjects instantiated from different files. Mappings considered in the GTFS-Madrid-Bench maximize the number of joins among the different files composing a GTFS feed. As for SQL queries, a *join* operation increases exponentially the number of comparisons needed with a growing number of items (in our case, the number of individuals instantiated in each file) resulting in a non-linear growth. In Section 3.2.4, we will showcase how *join*s in mappings can be avoided in some cases hugely improving performances.

2. ***Access to records***. The fact that CSV conversion time outperforms the JSON one is due to the performances of the library used in accessing the files. Also intuitively, accessing rows in a CSV file and iterating over them is less expensive than querying a JSON file resolving a

JSONPath query and iterating over nodes retrieved. This aspect affects the execution and worsens with the growing size of the file to be queried justifying the differences in timings obtained. As we will see in Section 3.2.5 this is also the main problem with XML files.

Results obtained showcase how materialization is a time-consuming approach considering large size datasets. However, it is important to highlight that the really good performances obtained in the lowering portion of the conversion are mainly attributable to the possibility of querying a materialized knowledge graph.

In Figure 39, we report times for the lowering portion of the conversion using a template-based approach. As expected, lowering times are equals both for CSV and JSON since the lowering portion is executed on identical knowledge graphs in the two cases. Moreover, we can observe how the growth of lowering times is sub-linear with respect to the scale growth, i.e. the lowering time of 5-csv is less than 5 times the one of 1-csv. This is mainly due to the optimizations implemented both in the query resolver (RDF4J library) and in the template engine (Apache Velocity).
The mentioned components help in obtaining good performances, however, in our experience with the lowerer, they can easily cause performance issues if not configured properly. In particular, two main aspects should be taken into account:

1. ***Optimize queries***. Access triples using simple queries and trying to avoid expensive constructs or patterns. It is better to divide complex queries into sub-queries, if possible.
2. ***Optimize template logic***. Template logic should avoid nested loops and support data structures (e.g. maps) may be used to access efficiently records in queries' result sets.

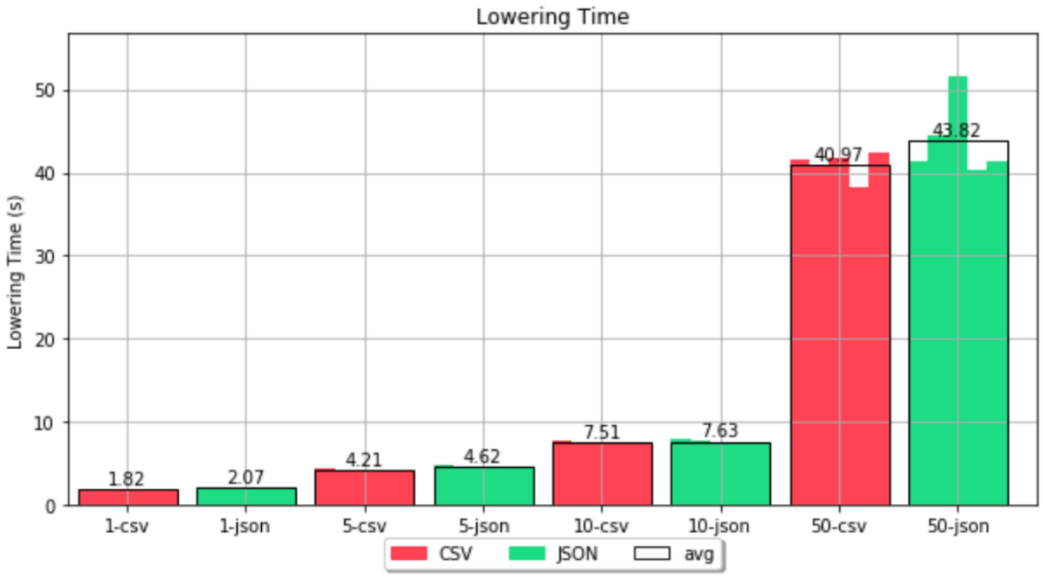


Figure 39: Lowering Time Chimera (Template Lowering) with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

Figure 40 and Figure 41 showcase, respectively, the CPU percentage usage and the max memory usage considering conversions executed.

CPU percentage usage is computed as user-mode-time plus kernel-mode-time, divided by the total elapsed-time. We can notice how CPU percentage usage is not influenced by the dataset size, and it is even smaller for higher size datasets probably due to the larger elapsed time considered that allows flattening peaks in CPU usage (shown in Figure 42) if the overall conversion is considered.

On the other hand, max memory usage is obviously highly affected by the dataset size since the materialized graph is stored using an in-memory repository. Whether it is possible to reduce the memory footprint of the conversion should be considered in the next release of the Chimera converter. Given the already presented results, it is possible to point out that time and not memory is the problem for higher size datasets (100,500-scale blocked for timeout), however, we will see in the next sections how more performant mappings can incur in out-of-memory problems and how Chimera is partially responsible for this problem.

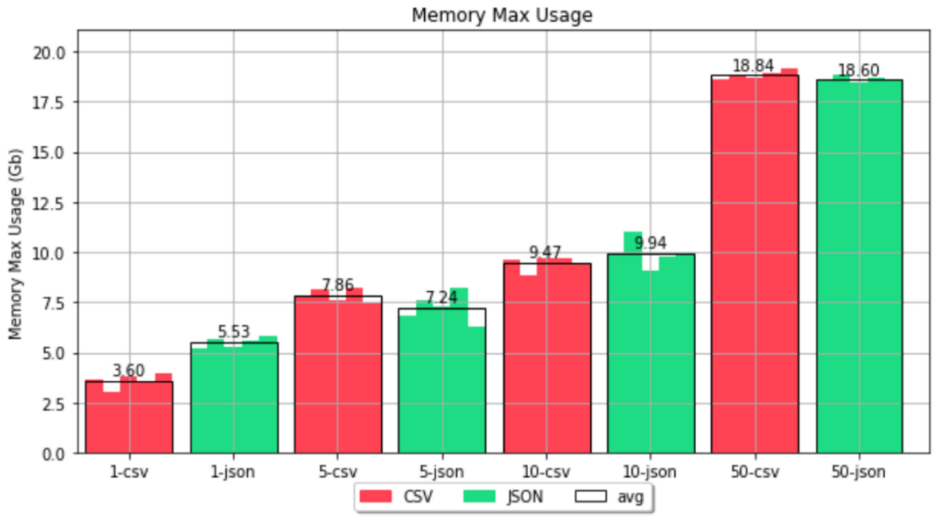


Figure 40: Max Memory Usage Chimera (RML-Mapper and Template Lowering) with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

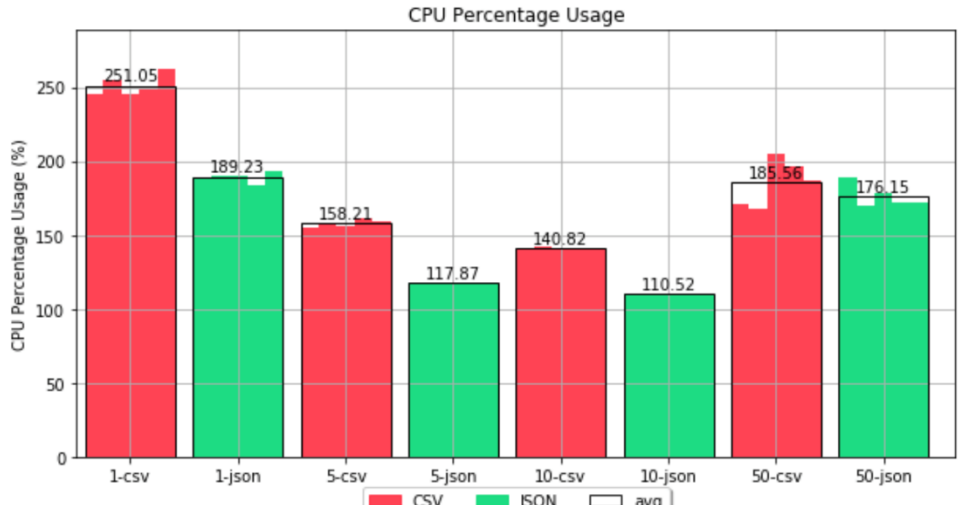


Figure 41: CPU Percentage Usage Chimera (RML-Mapper and Template Lowering) with 1,5,10,50-scale CSV and 1,5,10,50-scale JSON datasets.

Figure 42 and Figure 43 showcase the trend of CPU usage and Memory usage considering 3 repetitions of 50-scale CSV dataset conversion. It is possible to notice that both graphs have a peak near the end of the conversion and comparing timestamps it occurs at the end of the lifting phase. This is mainly due to two reasons. The first is related to the lifting block implementation, the RML-Mapper library stores the generated triples in an in-memory object that is returned at the end of the conversion. Then, the Chimera block copies triples in the shared RDF graph used in the pipeline doubling the memory required by the materialized knowledge graph in the copy operation. We identified this problem during the testing activities and we will modify the RML-Mapper library source code trying to solve the issue.

The second reason, but less important considering the graph at higher granularity, is related to the queries and processing performed in the lowering execution.

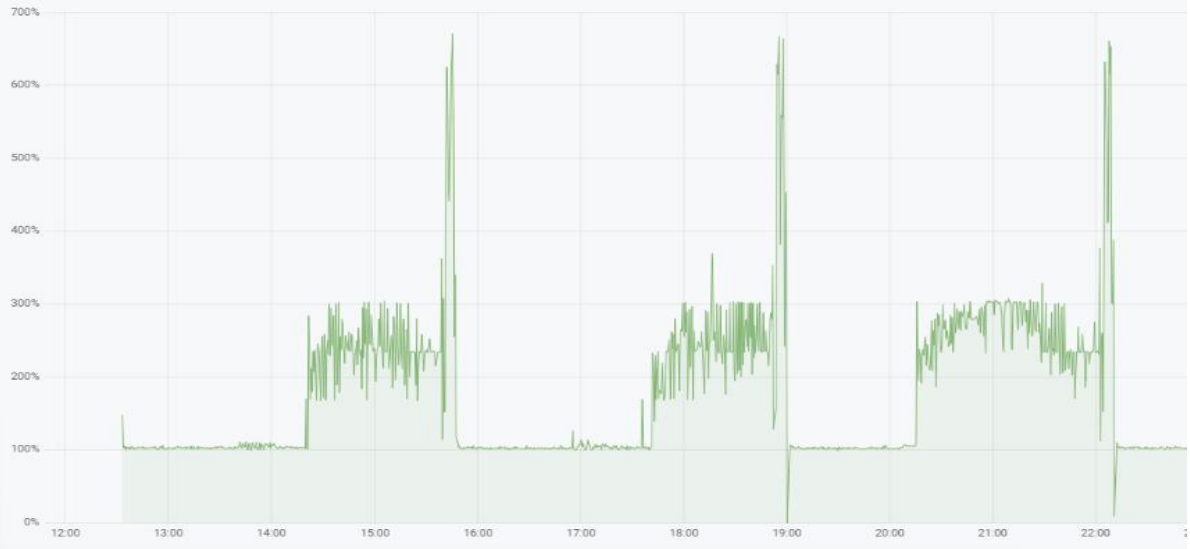


Figure 42: CPU usage Chimera during 50-scale CSV conversions (3 times).

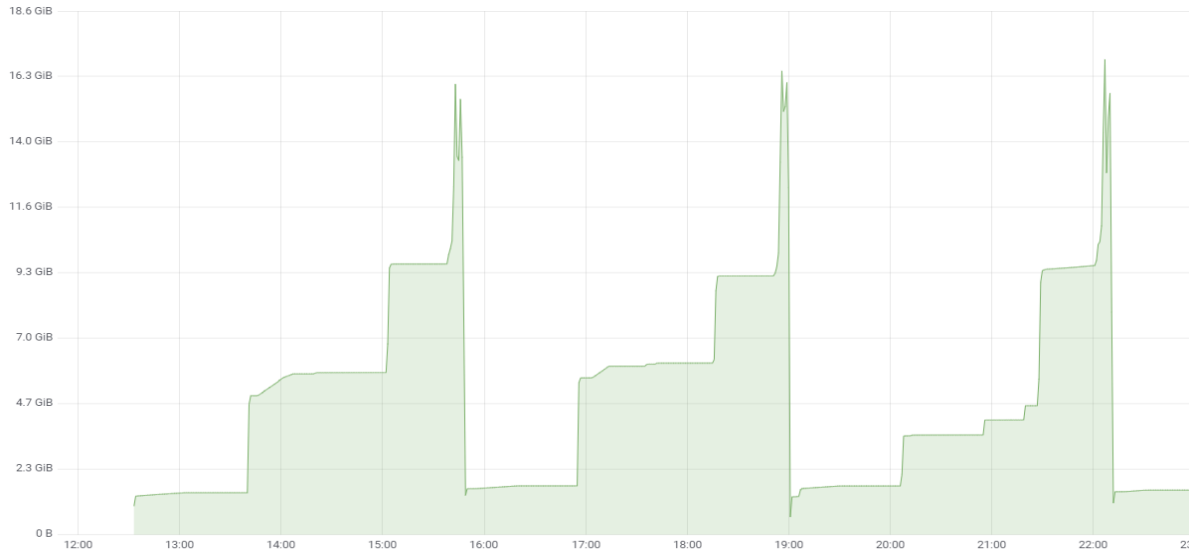


Figure 43: Memory usage Chimera during 50-scale CSV conversions (3 times).

3.2.3 Materialization Approach considering Batch Data with different RML mappers

In this section, to compare performances in the materialization, we report tests performed to compare the lifting block implemented in Chimera with *(i)* the RML-Mapper library to check if Chimera (using that library) introduced some overhead, and *(ii)* with the SDM-RDFizer tool (supporting RML mappings).

In Figure 45 we reported a graphical comparison of execution times for CSV datasets with scale 1,5,10 for the different tools. In Figure 44, we do not consider 50-scale since it abundantly exceeds other sizes considered. It is important to notice that in these graphs, the x-axis does not represent the dataset, but the results' set obtained for a given dataset using a given tool (the legend explain the relationships between colors and tools).

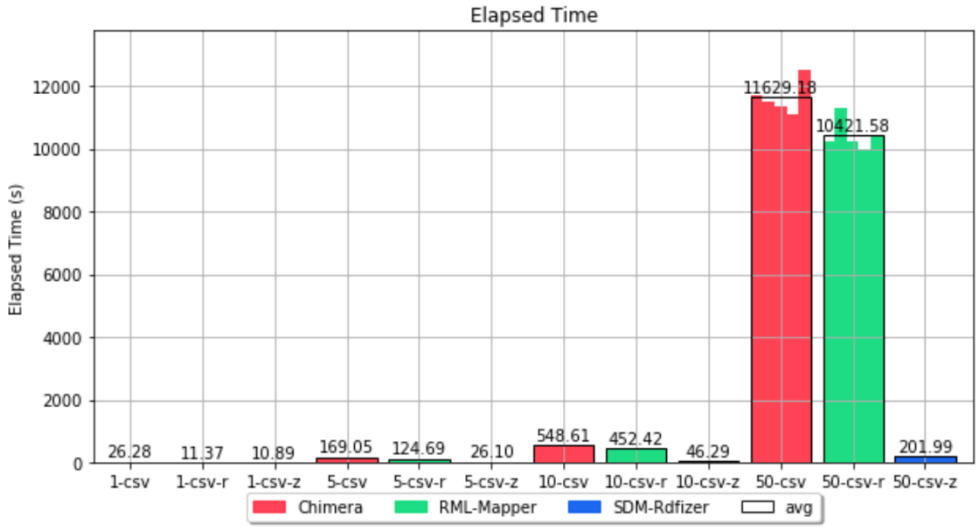


Figure 44: Elapsed Time for materialization considering different RML mappers with 1,5,10,50-scale CSV datasets.

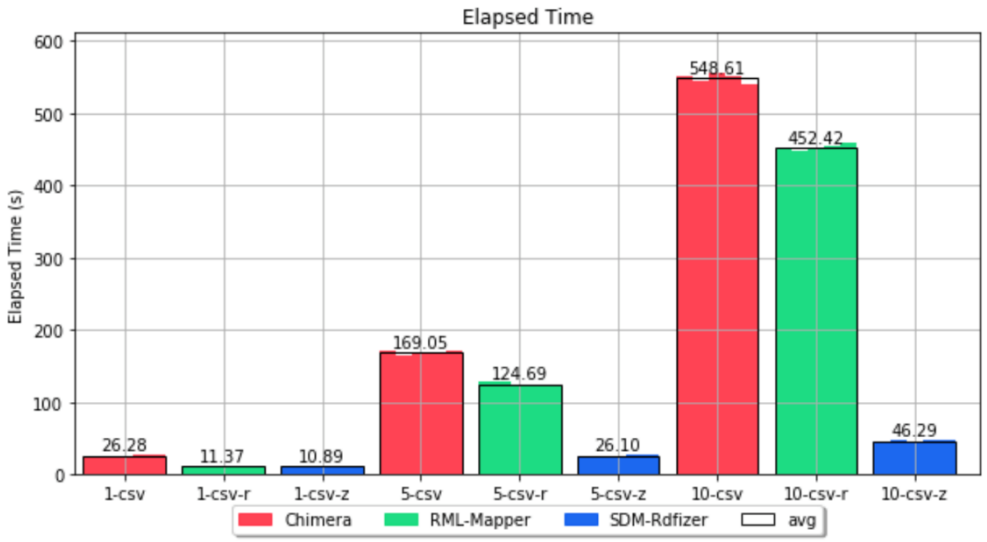


Figure 45: Elapsed Time for materialization considering different RML mappers with 1,5,10-scale CSV datasets.

In Figure 46 and Figure 47, we report data on CPU usage and max memory usage (respectively) considering the materialization for CSV datasets with scale 1,5,10,50 for the different tools.

Considering the RML-Mapper we may notice that performances are quite similar to the ones of Chimera. The main overhead introduced is related to what has been already pointed out in the previous section, i.e. to the copy operation of triples generated to the in-memory RDF graph used in the Chimera pipeline. This is the main bottleneck that should be addressed in the F-Rel implementation since it affects heavily the max memory usage.

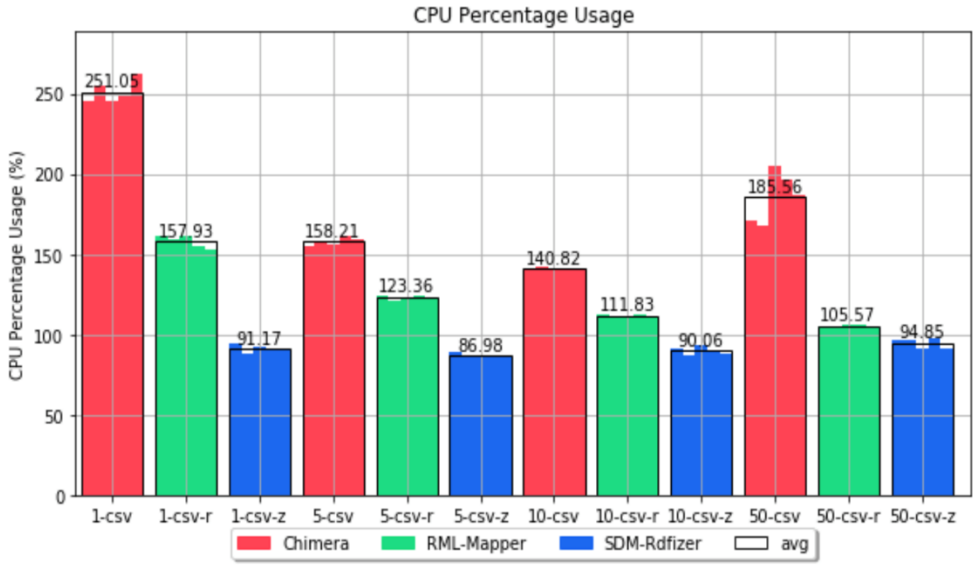


Figure 46: CPU Percentage Usage for materialization considering different RML mappers with 1,5,10,50-scale CSV datasets.

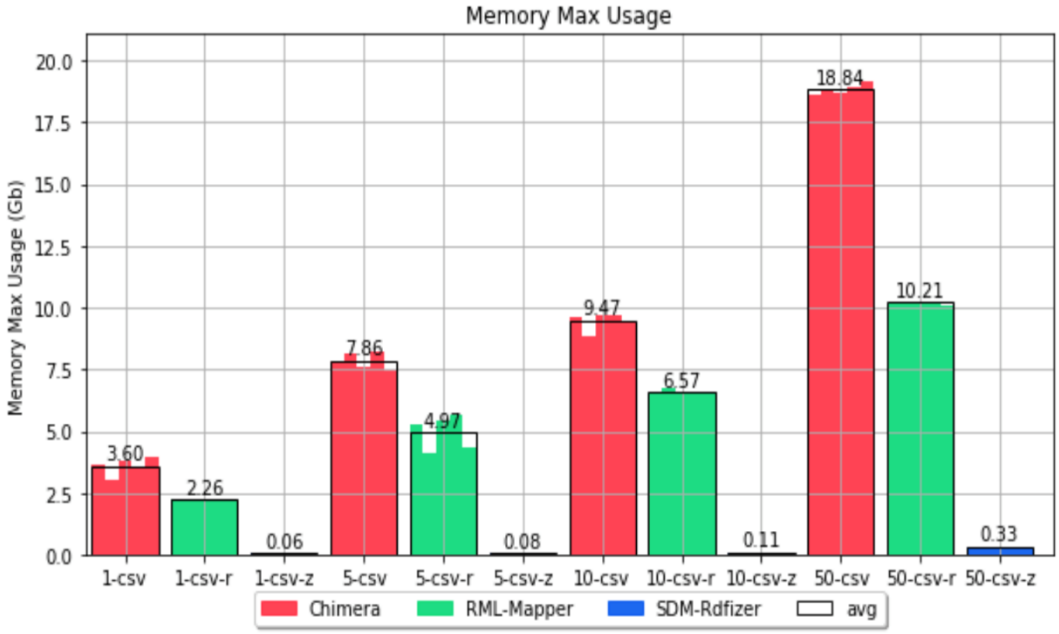


Figure 47: Max Memory Usage for materialization considering different RML mappers with 1,5,10,50-scale CSV datasets.

Considering the SDM-RDFizer results obtained showcase how its performances are much better of the one obtained using the RML-Mapper library, both in terms of execution time and max memory usage.

However, the SDM-RDFizer library has two main drawbacks. First, it is written in Python, making its integration with Chimera only possible as an exposed and callable external service (e.g. through a REST API). Second, it only deals with CSV data at the moment.

These two reasons still make RML-Mapper a preferable option for its integration in Chimera. Nevertheless, we will consider implementing a Chimera block dealing with an external service based on SDM-RDFizer to check performances considering a complete conversion integrating this tool for the lifting part. Moreover, the tests performed highlights the need to investigate further the logic behind SDM-RDFizer to understand if some of the implemented optimizations can be ported in the RML-Mapper library.

3.2.4 Materialization Approach using Optimized RML Mappings

In this section, we analyze how *joins* conditions in RML mappings can affect performances of the materialization. Before considering tests and results, we clarify in which cases and how it is possible to avoid *joins*.

In RML, to create a triple involving individuals instantiated in different Triples Maps[22] can be generated with two strategies:

[22] A triples map specifies a rule for translating each record extracted from the source file to zero or more RDF triples.

- *join condition*: specific RML construct to define the *join* condition determining the objects in the two data sources that should be linked by a specific property.
- *IRI*[23]: use information in one Triples Map's record to determine the *IRI* of the individual generated in another Triples Map and to be linked by a specific property.

While the first option is more generic and allows us to cover more cases, it is really expensive when it comes to conversion performance. For this reason, in the case of large files, it is better to opt for the second strategy, since a join condition would require comparing each individual generated in the first TriplesMap with each individual in the second TriplesMap. In the considered case from GTFS to Linked GTFS, for example, we can generate the IRI of instances of the *gtfs:Agency* class using the *agency_id* field in the agency.txt file of the GTFS feed. Then, considering the *routes.txt* we can avoid declaring a *join* condition on *agency_id* field to match the individual generated from the *agency.txt.* If the IRIs generated for individuals in the *agency.txt* are composed using the *agency_id* field, we can simply generate and refer the same IRI using the *agency_id* information present also in the routes.txt file. In the CSV case, this strategy can be applied only if the *join* condition is expected to build an *n-to-1* relationship, since a Triples Map accessing CSVs can access a row at a time. In our example, this is true since each *gtfs:Route* generated from the *routes.txt* file would refer exactly one gtfs:Agency generated from the *agency.txt* file. If it is not the case, sometimes it is possible to handle the problem similarly generating the inverse property, if available, and then inferencing the desired property (a *1-to-n* property has an *n-to-1* inverse property).

Using the explained approach, we managed to compose an optimized set of RML mappings for the GTFS-Madrid-Bench dataset generating a valid knowledge graph using the LinkedGTFS ontology and containing the necessary information to reassemble the original GTFS files in the lowering phase. All the *join*s condition were eliminated, except for the one related to *parent stations* in the *stops.txt* file (join referring individuals generated by the same triples map) that couldn't be replaced properly via IRI generation. Therefore, we used the optimized mappings to compare conversion time in Chimera with, and without *join*s conditions. To make a fair comparison, we slightly modified also the original mappings from the GTFS-Madrid-Bench to be sure that the materialized graph generated in the two cases had the same number of triples.

In , we reported the results obtained with Chimera, using the same pipeline with RML-Mapper lifting block and Template-based lowering block, for CSV datasets with scale 1,5,10,50,100,500. *J* represents the result obtained with original mappings maximizing the number of *join*s conditions, *O* represents the result obtained with optimizes mappings minimizing the number of *join*s conditions. TO stands for timeout (exceeding 24 hours), OOM stands for out-of-memory (exceeding the 24GB memory limit).

[23] IRI (Internationalized Resource Identifier) is the identifier of the individual generated

Table 14: Comparison Conversion Time considering different RML mappings within Chimera (RML Mapper and Template Lowering) with 1,5,10,50,100,500-scale CSV datasets.

	1		5		10		50		100		500	
	J	O	J	O	J	O	J	O	J	O	J	O
CSV	13.64s	12.03s	95.24s	51.31s	311.12s	144.22s	6205.25s	2269.62s	TO	OOM	TO	OOM

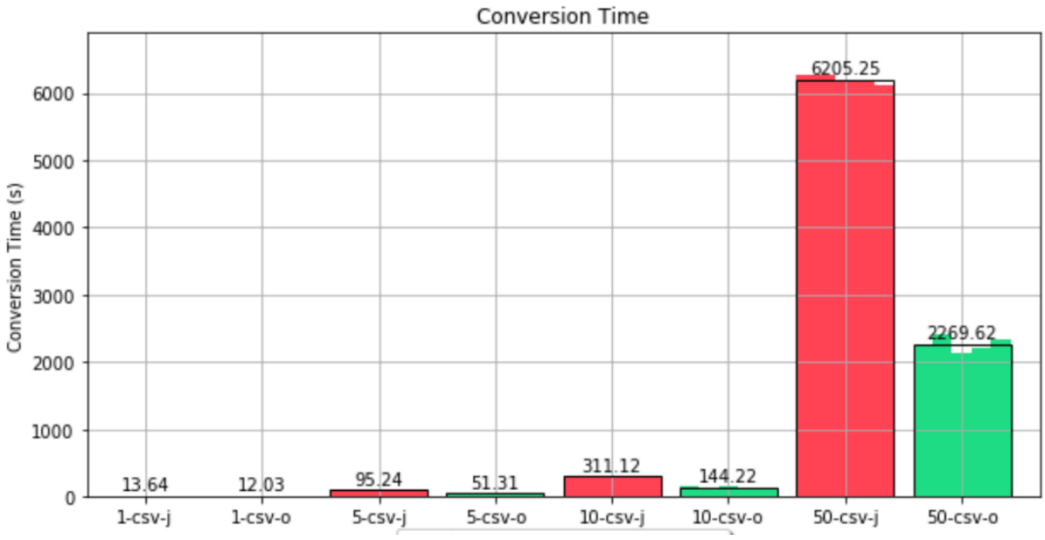


Figure 48: Conversion Time for conversion considering different RML mappings within Chimera (RML Mapper and Template Lowering) with 1,5,10,50-scale CSV datasets.

In Figure 48, we reported a graphical comparison of conversion times considering conversions terminated. In Figure 49 and Figure 50, we reported the CPU usage and max memory usage respectively.

It can be noticed that the *optimized* mappings reduce the conversion time required of up to 2/3 in the case of 50-scale CSV. On the other hand, however, CPU usage and max memory usage are not affected.

An important aspect is related to the fact that 100- and 500-scale datasets conversion failed due to memory reasons in this case. Therefore, considering also observations done in the previous sections, if we manage to reduce the memory overhead introduced by Chimera, we may be able to complete also these conversions within the timeout.

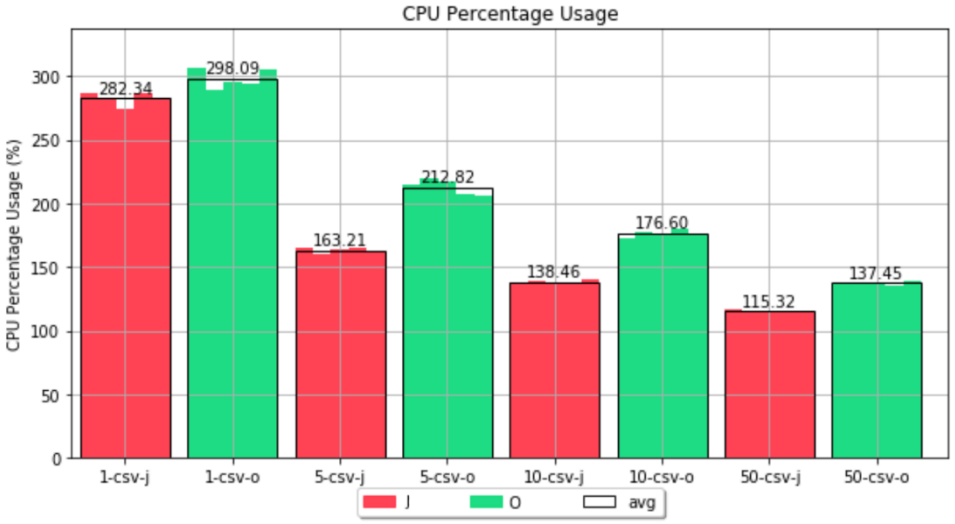


Figure 49: CPU Percentage Usage for conversion considering different RML mappings within Chimera (RML Mapper and Template Lowering) with 1,5,10,50-scale CSV datasets.

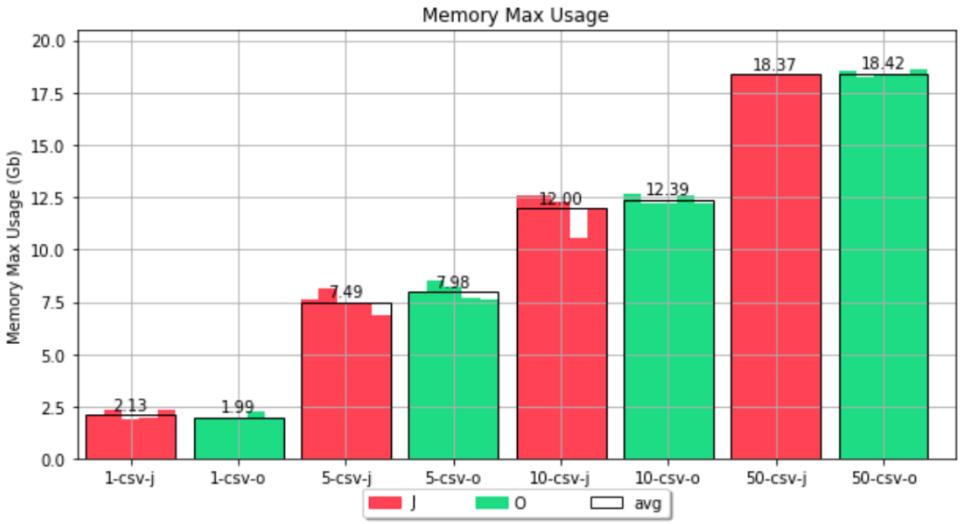


Figure 50: Max Memory Usage for materialization considering different RML mappings within Chimera (RML Mapper and Template Lowering) with 1,5,10,50-scale CSV datasets.

3.2.5 Materialization Approach considering Batch Data – The XML case

In this section, we present some tests specifically related to the conversion of XML datasets. As stated in Section 3.2.2, even the conversion with the 1-scale XML dataset reached the timeout (set to 24 hours) without completing the Chimera pipeline's stages. To validate the mappings and the functioning of the lifting block also for XML data sources we operated two different tests. First, we reduced the size of the 1-scale dataset to check whether the conversion finishes with a small amount of data and produces the expected result. Then, once checked this, we developed a set of optimized mappings, as the one described in Section 3.2.4, also for JSON and XML datasets to offer a comparison of performances of the lifting block on the three different data formats. In Figure 51, we report the conversion times obtained using Chimera (RML-Mapper for lifting with optimized mappings and Template-based lowering) considering 1-scale CSV, JSON and XML datasets. the Figure 52, we report only lowering times of the executed conversions.

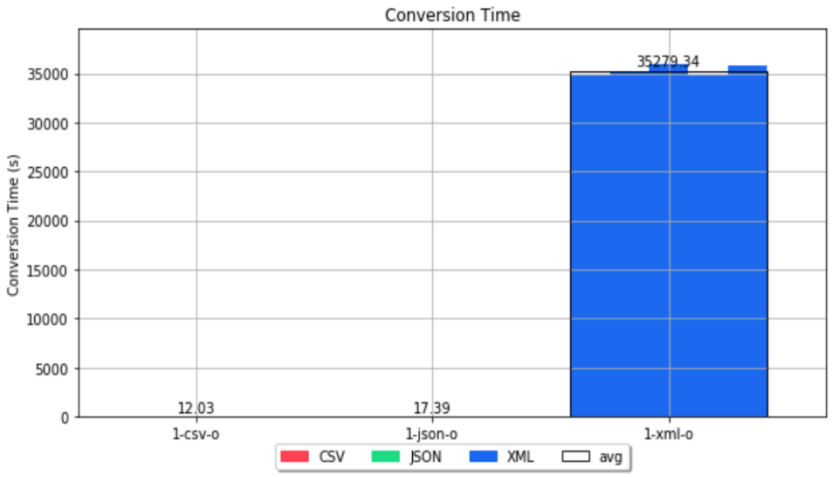


Figure 51: Conversion Time for materialization with Chimera (RML-Mapper and Template Lowering) considering optimized RML mappings with 1-scale CSV, JSON and XML datasets.

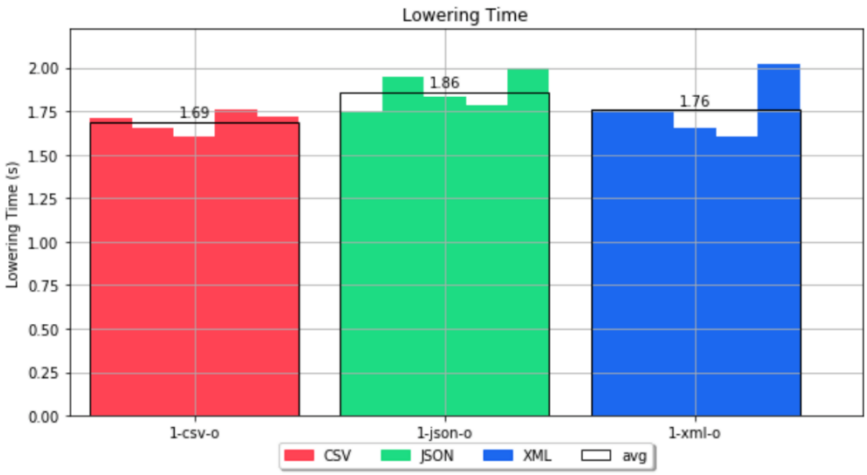


Figure 52: Lowering Time for materialization with Chimera (RML-Mapper and Template Lowering) considering optimized RML mappings with 1-scale CSV, JSON and XML datasets.

The graphs represented show how XML lifting performs much worse with respect to the CSV and JSON one, more than nine hours for XML and less than 20 seconds for CSV and JSON. The lowering times confirm that the conversion time is almost entirely due to the lifting stage and, as expected, they are similar for the different data formats given that the generated knowledge graph is identical in the three cases.

Figure 53 showcases how also memory consumption in the XML case is much worse with respect to the CSV and JSON case. We expected JSON and XML to consume more memory than CSV since the data structures to parse JSON and XML are more complicated, however, the XML consumption is almost the double with respect to the JSON case.

The bottleneck is known, and it is pointed out also by developers of the RML-Mapper library. The XML parsing implementation (*javax.xml.parsers*) to access XML files has been chosen to support full XPath, but it causes a large memory consumption (up to ten times larger than the original XML file size) and longer execution time.

It should be investigated how adopting a different XML parsing implementation may improve performances. Of course, changing the XML library will require to limit the expressivity of XPath queries in accessing the file records, but in several use cases, the full XPath specification is not required. Moreover, queries may be simplified adding a data preparation phase customizing the original XML file or building lighter XML files to support the lifting procedure.

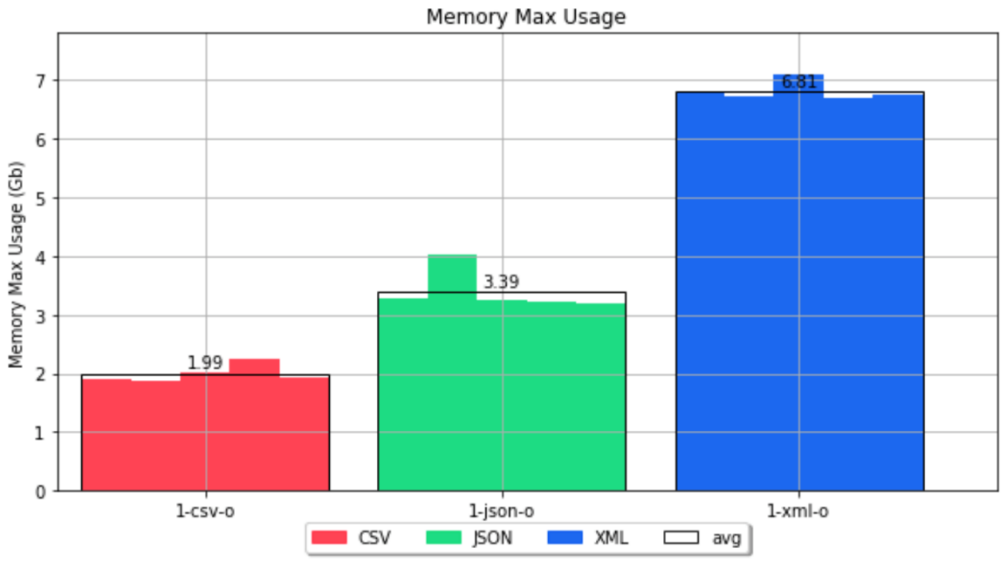


Figure 53: Memory Max Usage for materialization with Chimera (RML-Mapper and Template Lowering) considering optimized RML mappings 1-scale CSV, JSON and XML datasets.

3.2.6 Materialization Approach considering Runtime Data/Message Conversion

This section focuses on test cases for the Runtime data/Message conversion scenario. In this case, we do not expect large size datasets to be converted but small amount of data to be quickly converted to avoid adding significant overhead to the communication.

The ST4RT project specifically dealt with this problem proposing an annotation-based method to convert messages using a set of Java classes instrumented for the lifting and lowering mappings from/to the reference ontology through Java annotations.

As specified in the D5.2 deliverable, we integrated the engine developed in ST4RT as a lifting and lowering block in Chimera to exploit the modularity and the composability of the new architecture. Moreover, Camel offers a set of default blocks to expose REST APIs and to handle XML messages, thus simplifying and optimizing the codebase needed.

In our C-REL tests, we chose to consider the same use case analyzed in ST4RT to validate the correctness of the new implementation and to compare results obtained. The use case is described in D5.1 from the ST4RT project [13]. It imagines a scenario involving the SNCF's Reservation system using FSM and issuing an FSM Booking_PreBookRequest. This request should be forwarded to Trenitalia's reservation system (code 83) using 918 and therefore the converter should translate the request into a 918 uic_reservationbookrq. Then, the 918 uic_reservationbookrp reply from Trenitalia should be converted into an FSM Booking_PreBookResponse and sent back to SNCF.

In the ST4RT tests, the overall call was synchronous and Trenitalia processing of the request was simulated using the Boomerang tool. In our case, we consider separately the two conversions referencing the specific KPIs measured in ST4RT [14], i.e, the KPI 2 "Average translation of request duration" and the KPI 5 "Average translation of response time", composing two pipelines in Chimera to implement the two translations needed.

The testing environment considered for testing Chimera is similar to the one used in ST4RT, we run tests on a machine with the following specifications: Intel(R) i7-8565U (CPU @1.80GHz, 1992 Mhz, 4 Core(s), 8 Logical Processor(s)), 16.0 GB RAM, 1TB SSD. In our tests, we considered the different message types used in ST4RT and presented in Table 15 to check also if variations can influence the conversion process and affect in a relevant way the translation times.

Table 15: Messages used for the FSM/918 tests

Request ID	N° of passengers	Berth Type	Class
1	1	SINGLE	FIRST
2	1	DOUBLE	FIRST
3	1	SINGLE	SECOND
4	1	DOUBLE	SECOND
5	2	DOUBLE	SECOND

In Table 16 we present a comparison of results obtained in the ST4RT project for Pilot evaluation [15] and with Chimera for the considered KPIs. Each translation is run 3 times for each message type and also average values are reported. In our tests, time is measured as the time of the local API call sending the input message and receiving back the response.

Table 16: Comparison annotation-based conversion times in ST4RT and SPRINT implementations

	KPI 2 ST4RT	KPI 2 SPRINT	KPI 5 ST4RT	KPI 5 SPRINT
Request Id 1 - 1	4,00s	0,85s	4,70s	2,31s
Request Id 1 - 2	3,90s	1,17s	4,60s	2,36s
Request Id 1 - 3	4,30s	0,91s	5,30s	2,09s
Avg Request Id 1	4,07s	0,98s	4,87s	2,25s
Request Id 2 - 1	4,27s	0,99s	5,18s	2,27s
Request Id 2 - 2	3,80s	0,97s	4,60s	2,36s
Request Id 2 - 3	4,10s	1,05s	5,00s	2,24s
Avg Request Id 2	4,06s	1,00s	4,93s	2,29s
Request Id 3 - 1	3,89s	0,83s	5,33s	2,49s
Request Id 3 - 2	3,70s	1,10s	5,00s	1,95s
Request Id 3 - 3	3,70s	0,81s	4,50s	2,32s
Avg Request Id 3	3,76s	0,91s	4,94s	2,25s
Request Id 4 - 1	5,10s	0,87s	5,10s	2,02s
Request Id 4 - 2	5,10s	0,99s	4,50s	2,27s
Request Id 4 - 3	4,00s	0,93s	4,90s	2,05s
Avg Request Id 4	4,73s	0,93s	4,83s	2,11s
Request Id 5 - 1	4,80s	1,22s	5,50s	2,21s
Request Id 5 - 2	3,90s	0,98s	4,60s	2,40s
Request Id 5 - 3	4,10s	0,88s	4,80s	2,19s
Avg Request Id 5	4,27s	1,03s	4,97s	2,26s

As a first result, we validate the correctness of conversion pipelines implemented, in particular, testing activities carried out allows us to find and fix a bug in the pipeline implementing conversion from FSM request to 918 request.

Most importantly, results reported showcase how the SPRINT implementation outperforms the one used in ST4RT. As an initial observation, given that no modifications have been made to the engine, we think that this is mainly due to the fact that in Chimera we used the XML unmarshalling component

available in Camel. We will investigate further results obtained for F-REL also considering additional KPIs and different pipelines for the runtime data/message conversion scenario.

4. BUSINESS AND MARKET VALIDATION

This section is focused on the evaluation of the first proof of concept of the Interoperability Framework from business and market point of views. Indicators for the evaluation are based on the recommendations and for the IF development and deployment produced in GOF4R D5.2 “Toolkit of recommendations and KPI Scoreboard”.

Indicator	Description	How the proof of concept of the IF contributes to this indicator
Data openness	To what extent the solution supports data openness that influences market uptake of the IF positively, provided that the data opening does not negatively impact the provision of services operated under public service obligations.	Outcomes of previous project and actual implementation experience show that to be effective “data openness” must considered as an operational regime for digitalized business processes and not only as a particular design of data and data exchanges. The proof-of-concept provides tooling, implemented as the Asset Manager, for supporting a full set of business scenarios (S1-S9) that are all required to operate successfully with open data, including leveraging the availability of National Access Points (NAPs), particulary in MaaS implementations. Market uptake of the IF is facilitated inasmuch as it facilitates, in turn, the full exploitation of open data concepts and policies for the provision of advanced digitalized end-to-end multimodal mobility services.
Gaining the critical mass of IF participants	Easiness in joining the IF ecosystem which can lead to the increasing the number of users in the IF ecosystem	By minimizing or eliminating the need for adaptation of legacy systems and the need for participation in centralized govenance, joning the IF ecosystem is reduced to the registering/publication of own resources and the discovery of other stakeholder resources in the Asset Manager This process is analogous to publishing an own web page with links to web pages of other stakeholders in the world wide web: it is in fact this paradigm that has powered the spectacular seld-sustained growth of the web. Similarly, Joining the IF in this way provides access to many more resources that can be used to generate rich services to customers, creating a powerful self-sustained incentive to participation.

Market diversity and inclusiveness	Removing the distinction between big market players and small TSPs. IF's scope should be widened to MaaS operators. Ability to deal with different stakeholders' policies and regulations.	By minimizing ICT development costs and governance overhead, the IF tools eliminate barriers for access and participation in the interoperability ecosystem, particularly for smaller or specialist operators (e.g. providers of micro-mobility).
Stakeholder's management	One of the key elements that the IF governance has to adress: lack of cooperation among stakeholders, collaboration with other non-transport related entities (organizing authorities, financial institutions, IT services,…)	"lack of collaboration" is not an absolute but it is relative to a *specific* collaboration mechanism. Where this *specific* collaboration mechanism is based on mandatory adoption of data formats and the movement of large data sets according to a centralized calendar the 'lack of collaboration' is a reflection of the large overhead costs associated with controlling how these formats are kept common and how and when these data sets are moved across multiple systems. The IF is designed specifically to make differences across data formats irrelevant for application developers, removing the need for 'adoption' of one and the same for all, and to support but not mandate the movement of data sets from system to system. By making interoperability a digitalized mechanism instead of a 'governance' policy, the nature of the collaboration mechanism is changed into a task of registering/publishing own resources in the Asset Manager keeping full control of them, and discovering available resources registered /published by other stakeholders. The nature of governance is therefore also changed from enforcing 'collaboration' *policies* to maintaining the collaboration *tools*, i.e. the IF components themselves.
User-friendliness	To what extent the solution addresses the issue of lack knowledge in semantic ontology and data-interoperability	The solution uses standard semantic web specifications and other standard technologies, and is architected to allow for flexible composition of processing chains from common blocks through configuration scripts of open source build and runtime frameworks., minimizing the need for ICT development. Knowledge of these technologies and frameworks, and of their use in semantic interoperability, is necessary to develop new common blocks or features of the IF, but being

		standard it is generally available to ICT professionals. On the other hand, since the IF's purpose is to mask the machinery of interoperability from business application developers, the latter do not need to acquire, in general, specialist knowledge of the underlying technologies.
Reliability and security of the ecosystem	How the solution addresses the issues on cybersecurity.	The specific architecture chosen for implementation of the proof-of-concept separates provisions for reliability and security from the interoperability mechanism components so that the latter are deployable units of pure functionality while reliability and security are delegated to the runtime environment in which the components execute. In this way a specific runtime environment can be configured for specific reliability and security requirements, protecting components that do not need to implement their own and can therefore be standardized.
Creation of new business models	How the solution contributes to creation of new business models (e.g., MaaS), costs and/or benefits incurred between different modes of transport	The main purpose of the IF is to mask the machinery of interoperability from business application developers, in the same way that a computer operating system masks the machinery of internetworking from application users or developers. In this sense the IF does not incorporate *application level* requirements and for this reason does *not* induce dependencies on the range and type of applications that can be developed. There is in fact no justification for different interoperability mechanisms for different transportation modes even if data formats may be specific for each mode for historical or other reasons: by creating a common knowledge graph that can be queried semantically from any data format or data source the IF provides a common abstraction of all data resources of all modes so that a whole new range of new business level applications can be created.
Costs to join the IF	How the solution addresses the issues of costs associated with the introduction of the IF (acquiring the skills of	By separating interoperability mechanisms (implemented in IF components) from business application development (which is not concerned with interoperability) semantic interoperability skills are only necessary for developing new

	semantic interoperability and adaptation of legacy systems).	features, e.g. new converters, of the IF, but not for business application developers, e.g. travel service providers. Also, legacy system adaptation is minimized or altogether eliminated by semantic, i.e. automated, conversion across hetereogeneous data specifications, as demonstrated already by the previous IT2RAIL and ST4RT projects.

5. ADDITIONAL REQUIREMENTS FOR F-REL

In this section, we discuss the functional validation presented in Section 2, and the performance and scalability evaluation described in Section 3, identifying additional requirements for F-Rel.

5.1 FUNCTIONAL VALIDATION

The functional evaluation of how the Asset Manager can be used to implement our scenario showed that all of the scenarios can be fully implemented using the tool. Anyway, in F-Rel some of the features can be better shown to the user. In particular, the Store application is currently able to render metadata of assets which are directly managed by the Asset Manager. While integrating the Distributed SPARQL endpoint, we found out that the Asset Manager can be used also to integrate different catalogues and provide an unified interface for accessing them. The F-Rel version of the Store could be modified to achieve such result, and distinguish between locally-managed assets which are published and managed via the BPMN lifecycle processes deployed on the Asset Manager, and remotely-published assets which come from remote (but trusted) sources. Such feature would enable integrate a single Asset Manager with different National Access Points, and that would enable a pool of companies using the SPRINT IF to use the Asset Manager as a single point for accessing transport data pertaining to different countries, and to re-use such data in a controlled way.

The C-Rel version of the User Manager is a modified version of the authentication and authorization layer provided by Django and its libraries (which provide JWT and asset-level permissions). In its current status, the User Manager is therefore “embedded” inside the Asset Manager, and therefore without an Asset Manager there cannot be authentication and authorization functionalities inside a deployment of the IF. The final version of the SPRINT IF should investigate the usage of off-the-shelf, open source solutions for identity providers. With an independent Identity Provider, the SPRINT project could leverage on widely accepted standards (like OAuth 2.0) for authentication and authorization, and maybe also leverage on the “federation” features that could enable accessing different IF nodes with the same user credentials.

The initial functional validation of the Mapping Tool is promising[24]. However, the tool needs to be extended/improved in several ways. In particular the enhanced (F-REL) version of the Mapping Tool will have some added features. Currently, the Mapping Tool is a command-line application, but a Graphical User Interface will be developed for the tool to increase its user-friendliness. The final release of the tool will also offer an automated generation of annotations, which are necessary for the conversion mechanism used by Converter components of the IF.

The mapping techniques also need some improvement. Indeed, the examples used for the evaluation were relatively simple, in the sense that terms in the source and target standards were in many cases identical (for the evaluation, we chose standards for which we could reasonably check in a manual manner the fitness of the suggested mapping, which limited the range of standards that we could use). Hence, in F-REL we aim to improve the mapping techniques to achieve an acceptable accuracy for a greater range of pairs of standards, from similar ones – i.e., pairs of standards with

[24] The accuracy of the Mapping Tool varies depending on the standard: the maximum accuracy is 76%, while the minimum is 67%, which leads to an average accuracy 72%.

the same roots – to more heterogenous ones – standards that have fewer identical terms even though the information that the cover is similar.

5.2 Performance and Scalability Evaluation

We have introduced the preliminary performance and scalability tests performed to evaluate artifacts developed for C-Rel in Section 3. This section analyzes the results obtained and discuss developments for FREL considering two critical aspects for the IF and the related components, i.e., querying and converting heterogeneous data.

5.2.1 Querying Heterogeneous Data

Querying data is a task which is related to two different tasks in the context of SPRINT components. The former one, also according to the IF architecture and the scenario S4, is the execution of distributed queries on the metadata catalogue in the Data Layer. To be able to fulfill the requirements implied by scenario S4, we let the Asset Manager interact to the Distributed SPARQL endpoint. The second task where querying is important is during the conversion of messages and datasets. In this second context, it is not only important to perform distributed queries, but it is also important to access data which is not natively in RDF format. The task to be performed is therefore much more complex, because it involves translating queries from SPARQL into the specific query language supported by the different data sources.

For the C-REL, we have tested the performance and scalability of the Distributed SPARQL endpoint as an isolated IF component. Since the main implementation that we've chosen supports both distributed querying and accessing virtual knowledge graphs, we were able to test both of the aforementioned usage scenarions. Monitoring components in isolation allows us to easily detect potential bottlenecks in IF architecture components and also facilitate the testing phase by creating a simpler environment for testing a component. In this way, we have created a test suite using Docker and recreated a distributed environment for the Distributed SPARQL endpoint.

As described in D3.3, the amount of metadata required by the Asset Manager is fairly low and the metadata schema does not allow for very complex SPARQL queries. However, we can get more accurate measurements and conclusions statistically when the volume of data is high and the query structure is varied. Therefore, we have decided to test the Distributed SPARQL endpoint on datasets with real data from the public transport sector (and not with metadata) and with queries of different complexity. We decided not to ship Ontario with the default installation of the Asset Manager due to license incompatibilities. Ontario is licensed under the GNU/GPL v2, while the Asset manager (as also Chimera) is licensed under the Apache 2.0.

In our tests, we considered different query engines that provide a unified virtual representation over different data sources in several formats. Among the engines that are capable of performing distributed queries on data sources are Ontario and Squerall. However, we have not reported performance results with Squerall because we have not found a way of how to start this engine and there is no comprehensive documentation. In addition, we have compared Ontario with other non-distributed engines such as Morph(-RDB/CSV/xR2RML) and Ontop. Based on the results obtained, we can say that the virtual engines still need more research before being part of a production system since in most cases they fail because: i) they do not fully support SPARQL 1.1 operators; ii) they produce incorrect or incomplete responses, i.e., the number of results obtained differs with respect

to the baseline (RDF materialised graph); iii) the performance and scalability is very low when data size increases.

Based on the results of the performance and scalability of the Distributed SPARQL endpoint by using our benchmark, we consider the experimentation of the Distributed SPARQL engines to access non-RDF sources to be over, since they require internal improvements in their operation that depend on each of the development teams and it is not an affordable task to program a new query translation engine from scratch in order to give support to all the data formats that can be uploaded to the asset manager. The usage of the Distributed SPARQL endpoint to access remote RDF repositories in conjunction with the Asset Manager will instead be continued, and we will carry on a better integration of the two components to better address the requirements of letting Asset Manager users access consolidated metadata coming from both the internal Asset Manager storage and remote metadata catalogues such as open data portals and National Access Points.

The main objective of the proposed benchmark is not to provide a ranking of engines, but also to provide a set of resources that can be useful for: (i) practitioners to choose the engine that fits better for their use cases and (ii) developers of query translation engines to improve their tools and compare their results with other proposals. As such, we expect this benchmark to be a stepping stone in this area where much research and development has been done for decades, but there is a need for more mature applications to be used in real-world environments. Indeed, our experimental study has shown that there are still relevant open issues such as SPARQL conformance, semantic preservation in the translation from SPARQL queries to the query languages used to query raw data (CSV, JSON, XML), and the application of query evaluation optimisation techniques. With this proposal we intend to contribute to the community by providing not only the baseline that can be used to improve the development of the current engines, but also the possibility to use it to test new approaches and techniques over the next years. For these reasons and although a virtualization approach can be integrated during the lowering phase to implement a conversion pipeline by solving SPARQL queries in the lowerer template, we should investigate other manners to improve data transformation for FREL because there is still a lot of research to be conducted in the current field of construction of virtual knowledge graphs. Therefore, in the F-REL, and in order to improve the lowering phase, we will explore other ways of optimizing the materialization of the knowledge graph during the lifting and lowering phase in the conversion pipeline adapting ideas and techniques presented in virtual approaches.

However, to continue working on SPARQL queries for F-REL, we propose to include user preferences to the queries as proof of concept to check if some value can be added to the queries in the IF architecture. Skyline queries are a kind of queries based on user preferences that identify the set of rows that are not dominated by any other row, where a row is considered to dominate another one if it is as good or better in all criteria and better in at least one criterium [16]. Skyline queries can be used in transport applications to make better decisions when multiple criteria over data are specified. For example, a skyline query can be one that identifies which are the least used stops within the lines with the most demand.

5.2.2 Converting Heterogeneous Data

Considering the *batch data conversion* scenario, we tested performances and scalability of a Chimera pipeline for conversion implemented using the RMLMapper lifting block and the Apache Velocity Template-based lowering block. The presented analysis showcases the potentiality of the

implemented solution, managing to generate and handle knowledge graphs with millions of triples within the conversion pipeline. On one hand, we highlighted important aspects to be considered to optimize performances of both lifting and lowering approach. Most importantly, we showcased how the number of *join*s condition in the RML mappings highly affects performances. On the other hand, considering scalability issues we noticed that memory consumption can be a problem and Chimera introduces a not negligible overhead. One of the elements which contributes to the high memory consumption is the fact that all the data is processed in memory. For this reason, for F-Rel we will try to optimize memory usage in Chimera modifying the RMLMapper (e.g. trying to integrate an external repository to store the materialized graph) and studying the possible integration of additional blocks (e.g., a block interacting with the tested SDM-RDFizer or last optimizations techniques that have been proposed in the state of the art [17]). Moreover, since materialization is the main responsible of memory consumption, and considering tests performed in Section 3.1, we plan to test and compare conversion pipelines exploiting a hybrid solution including ideas defined in virtualization for the lowering phase.

Considering the *runtime data/message conversion* scenario, we tested and compared performances of the ST4RT annotation-based method ported in the new converter architecture (Chimera). Results obtained show a great improvement in performances. For F-REL, we will investigate further performances considering scalability in the number of concurrent requests made to the converter in this scenario. Moreover, considering *runtime data/message conversion* requirements, we will investigate performances and scalability of a declarative approach based on RML mappings and lowering templates, as the one used for the *batch data* scenario.

6. ANNEX A: REQUIREMENTS TRACEABILITY MATRIX

Requirements related to C-Rel version of the SPRINT Interoperability Framework have been described in several deliverables. This section consolidates them in a single list.

6.1 HIGH-LEVEL IF REQUIREMENTS

This section summarizes the high-level IF requirements discussed in D2.2. Table 17 lists all the requirements.

Table 17: Index of Data, Service and System Management Requirements

Data Management Requirement	
DR1	Data standardization and portability
DR2	Data accessibility and openness
DR3	Dataset lifecycle management
DR4	Depth of data
DR5	Efficiency
DR6	Machine readable data
DR7	Preservation of information
DR8	Quality of data
DR9	Security and Privacy
Service Management Requirement	
SeR1	Dataset publication and subscription services
SeR2	Discoverability
SeR3	Inclusion and accessibility for all types of users
SeR4	Multilingualism
SeR5	Quality of Service
SeR6	Service Efficiency
SeR7	Service Standardization
SeR8	User activity monitoring
SeR9	User-centricity of service design and implementation

System Management Requirement	
SyR1	Administrative simplification
SyR2	Authorization and Authentication mechanisms
SyR3	Fault tolerance and backup / Recovery mechanisms
SyR4	Guidelines
SyR5	Integration of complementary services
SyR6	Interoperable and flexible laws and regulations
SyR7	Profit vs Cost ratio
SyR8	Reusability
SyR9	Scalability
SyR10	Subsidiarity and proportionality
SyR11	System Efficiency
SyR12	System monitoring and assessment
SyR13	Technological neutrality and system/infrastructure harmonization
SyR14	Transparency

The following requirements have been selected as highly important for the IF:

- **DR1.Data Standardization and Portability**. In general, it aims at the harmonization of data specifications and representation formats, the unification of data communication protocols/interfaces and the convergence of database models and systems. This, in in turn, makes data coming from various systems portable and compatible with other systems, and leads to an interoperable ecosystem.
- **DR2.Data Accessibility and Openness**. It highlights the importance of encouraging and practicing free access to data. In general, Data Accessibility and Openness include two concepts, namely Legally and Technically open data. The former refers to increasing the accessibility of data by placing them in the public domain with minimal restriction, while the latter means that data should be openly discoverable, assessable, processable, and re-usable.
- **DR9.Security and Privacy.** In general, it refers to the requirement of keeping data safe and secure, and to make each piece of information only available for authorized entities.
- **SeR1.Service Efficiency**: How well a service utilizes available resources.

- **SeR7.Service Standardization**: It refers to the need to develop and publish services that adhere to some standard to ease their invocation.
- **SyR13.Technological neutrality and system/infrastructure harmonization:** As the name suggests, this requirement highlights the lack of standardization in the lower layer of the technology stack and the need to decoupling the services/functions provided by a system from the underlying enabling technologies.
- **SyR4.Guidelines:** This requirement covers two different categories of audiences: firstly, end-users, through the provision of comprehensive instructions for them to engage with the system; secondly, business partners, potential followers and any interested party who might enhance the system in future, through the provision of generic rules and recommendations to facilitate and direct them.

6.1.1 Non-functional requirements

Given that functional requirements are usually sets of functions/services that must be delivered by a system, we can define various non-functional requirements that must be satisfied by each. Table 18 shows the non-functional implications of all the identified functional requirements.

Table 18: Non-functional implications of some of functional requirements in each viewpoint

	Elicited Functional Requirement	Non-Functional Implication		
Data Viewpoint	DR3.Data Life Cycle Management	User friendliness	Reliability	Manageability
	DR7.Preservation of information	Reliability	Availability	Fault tolerance Back up
	DR6.Machine readable data	Durability	Efficiency	Robustness
	DR1.Data Standardization	Efficiency	Effectiveness	Maintainability
Service Viewpoint	SeR8.User Activity Monitoring	Reliability	Availability	Scalability
	SeR2.Dataset publication and subscription services	Availability	Scalability	Accessibility
	SeR7.Service Standardization	Effectiveness	Maintainability	Documentation
System Viewpoint	SyR5.Integration of complementary services	Efficiency	Scalability	Quality
	SyR12.System monitoring and assessment	Maintainability	Reliability	Effectiveness
	SyR4.Guidelines	User friendliness	Supportability	Usability
	SyR13.Technological neutrality and system/infrastructure harmonization	Efficiency	Scalability	Quality

6.2 FUNCTIONAL REQUIREMENTS OF SPRINT IF COMPONENTS

This section summarizes the requirements for each of the components being developed for the SPRINT Interoperability Framework, and provides a summary which requirements are already fulfilled in C-Rel and which requirements will be further investigated during F-Rel. For each requirement we will provide a link to the corresponding scenario in D5.1.

Requirements are arranged in tables for each component. Below you can find the explanation of the columns:

- ID: requirement identifier
- Description: requirement description
- Scenario: where the requirement can be verified. It can be either:
    - a Scenario as described in D5.1 (D5.1 S*x*)
    - “CONN” when the requirement comes from the collaboration scenarios between SPRINT and CONNECTIVE
    - “Dx.x” if the requirement is described in that deliverable.
- C-Rel status:
    - OK: the requirement is fully fulfilled
    - NO: the requirement is not fulfilled
- F-Rel plan:
    - YES: this requirement will be further studied in F-Rel
    - NO: this requirement will not be further studied in F-Rel

6.2.1 Ontology engineering tools

Table 19: Functional requirements for SPRINT Ontology engineering tools

ID	Description	Scenario	C-Rel status	F-Rel plan
OE-R1	Selection of XSD input reference data to be transformed	D4.2	No	YES
OE-R2	XSD2OWL must be able to allow generating an ontological file as output	D4.2	No	YES
OE-R3	XSD2OWL must be able to allow extract complex and simple types from XSD files	D4.2	No	YES
OE-R4	Collaborative ontology engineering must be able to documentation Generation of the selected ontologies	D4.2	No	YES
OE-R5	Collaborative ontology engineering tool must be able to evaluation the Generation of the selected ontologies	D4.2	No	YES
OE-R6	Collaborative ontology engineering tool must be able to access a public GitHub repository	D4.2	No	YES
OE-R7	Collaborative ontology engineering tool must be able to preview generated documentation	D4.2	No	YES
OE-R8	Collaborative ontology engineering tool must be able to generate documentation from a select step	D4.2	No	YES
OE-R9	Collaborative ontology engineering tool must be able to generate version control(using Git)	D4.2	No	YES

6.2.2 Mapping suggester

Table 20: Functional requirements for the SPRINT Mapping Suggester

ID	Description	Scenario	C-Rel status	F-Rel plan
M-R1	It must allow users to upload to the system standard/specification vocabularies in various formats including XML and XSD for source standard, and OWL and TTL (for the target ontology)	D5.3, S7	YES	YES
M-R1	It must build mapping of the concepts presented in the two input standards employing machine learning approaches.	D5.3, S7	YES	YES
M-R1	It must build mapping of the concepts presented in the two input standards employing both structural mapping and machine learning approaches.	D5.3, S7	NO	YES
M-R1	It must allow users to view the suggested mappings of the concepts.	D5.3, S7	YES	YES
M-R1	It must allow users to interactively review and choose between various suggestions through dedicated GUI.	D5.3, S7	NO	YES
M-R1	It must build Java and RML annotations based on the approved suggestions.	D5.3, S7	NO	YES

6.2.3 Distributed SPARQL endpoint

Table 21: Functional requirements for the SPRINT Distributed SPARQL Endpoint

ID	Description	Scenario	C-Rel status	F-Rel plan
DSE-R1	Allows queries on metadata catalog	-	Ok	NO
DSE-R2	Distributed SPARQL Endpoint that can receive and processing SPARQL Protocol requests	-	Ok	NO
DSE-R3	Allows you to obtain results from a query over different sources (metadata catalog)	D5.1 S4	OK	NO
DSE-R4	Distributed SPARQL Endpoint you to add more data sources	D5.1 S4	OK	NO
DSE-R5	Allows to process queries with preferences about the metadata catalog	D5.3 S4	NO	YES

6.2.4 Asset Manager

Table 22: Functional requirements for the SPRINT Asset Manager

ID	Description	Scenario	C-Rel status	F-Rel plan
AM-R1	The Asset Manager must be able to host an arbitrary number of asset types.	-	OK	NO
AM-R2	The Asset Manager must be able to host an arbitrary number of assets of a given asset type.	-	OK	NO
AM-R3	The Asset Manager must allow describing remote resources/assets.	D5.1 S6	OK	NO
AM-R4	The Asset Manager must allow attaching files to assets.	D5.1 S5	OK	NO
AM-R5	The Asset Manager must allow users to download attachments of an asset.	D5.1 S5	OK	NO
AM-R6	The Asset Manager must be able to visualise metadata of an asset	D5.1 S3	OK	NO
AM-R7	The Asset Manager must be able to link an asset type to a governance process.	-	OK	NO
AM-R8	The Asset Manager must be able to host an unspecified number of governance processes.	-	OK	NO
AM-R9	The Asset Manager must allow both human tasks and automatic tasks to be specified in a governance processes.	-	OK	NO
AM-R10	The Asset Manager must allow browsing the assets belonging to a specific asset type.	D5.1 S3	OK	NO
AM-R11	The Asset Manager must allow searching for a specific asset.	D5.1 S3	OK	YES
AM-R12	The Asset Manager must separate the Consumer's user interface and the Contributor's user interface.	D5.1 S1, S2	OK	NO
AM-R13	The Asset Manager must implement a "Request for access" process allowing consumers to ask for the	D5.1 S3	OK	NO

	permission to access and use an asset owned by a different user.			
AM-R14	The Asset Manager must hide the details and attachments of an asset if not owned by the user.	D5.1 S3	OK	NO
AM-R15	The Asset Manager must implement a notification system to inform users about the results of a request.	D5.1 S3	OK	NO
AM-R16	The Asset Manager should notify users via email.	D5.1 S3	OK	YES
AM-R17	The Asset Manager must allow publishing “Converter” assets.	-	OK	YES
AM-R18	The Asset Manager must allow publishing “Ontology” assets.	-	OK	YES
AM-R19	The Asset Manager must allow publishing “Mapping” asset.	-	OK	YES
AM-R20	The Asset Manager must allow publishing “RDF Dataset” assets.	-	OK	YES
AM-R21	The Asset Manager must allow publishing “Journey planning service/dataset” assets.	CONN	OK	YES
AM-R22	The Asset Manager must allow publishing “Booking service” assets.	CONN	OK	YES
AM-R23	The Asset Manager must allow publishing “Trip tracking service” assets.	CONN	OK	YES
AM-R24	The Asset Manager must allow publishing “Issuing service” assets.	CONN	OK	YES
AM-R25	The Asset Manager must allow specifying source and target specification/standard when publishing Converters.	D5.1 S8	OK	YES
AM-R26	The Asset Manager must allow both describing a Converter as a remote service and as a downloadable artifact.	D5.1 S8	OK	YES
AM-R27	The Asset Manager must allow reusing Ontologies, RDF Datasets and Mappings while describing a Converter.	D5.1 S8	OK	YES

AM-R28	The Asset Manager must support the execution of arbitrary post-processing scripts to support Continuous Integration and Delivery	D4.2, D5.1 S8	OK	NO
AM-R29	The Asset Manager must be able to create deployable artifacts for Converters, reusing the specified Ontologies, RDF Datasets and Mappings and assembling a conversion pipeline using the Converter Frameworrk (Chimera)	D5.1 S8	OK	YES
AM-R30	When creating Converter deployable artifacts, the Asset Manager must support deployment of a Converter as a single JAR.	D5.1 S8	OK	NO
AM-R31	When creating Converter deployable artifacts, the Asset Manager must support deployment of a Converter as a container.	D5.1 S8	OK	NO
AM-R32	When creating Converter deployable artifacts, the Asset Manager must support deployment of a Converter as an containerised application composed by a load balancer and a replicable/scalable conversion service.	D5.1 S9	OK	YES
AM-R33	The Asset Manager must support the publication of parametric SPARQL queries (“Exploration API”)	D4.2	OK	NO
AM-R34	The Asset Manager must allow accessing parametric queries as APIs with parameters	D4.2	OK	NO

6.2.5 User Manager

Table 23: Functional requirements for the SPRINT User Manager

ID	Description	Scenario	C-Rel status	F-Rel plan
UM-R1	The User Manager must provide a login mechanism	S1, S2	OK	NO
UM-R2	The User Manager must be able to assign different roles to users	S1, S2	OK	NO
UM-R3	The User Manager must be able to assign permissions to users and roles	S1, S2	OK	NO
UM-R4	The User Manager must be able to provide JWT tokens for API	-	OK	NO
UM-R5	The User Manager must provide a registration form which lets the user state the desired role	S1, S2	OK	NO
UM-R6	The user must be notified when he's successfully registered	S1, S2	OK	NO

6.2.6 Converter

Functional requirements for the IF Converter have been gathered considering the outcomes of the ST4RT project and the state of the art discussed in D4.1, the scenarios defined in D5.1, and the automation workflows described in D4.2.

The functional requirements identified have been completely fulfilled in the C-Rel implementation and only a few will be investigated further. In particular, we need to improve the framework for the service mediator use case to facilitate the creation of pipelines like the one discussed in Section 2.10 of this deliverable, and we will produce deployment templates for the IF Converter to execute it also on orchestration systems like Kubernetes.

The F-Rel development for the Converter will be driven by additional requirements discussed in Section 5.2.2 as a result of the performance and scalability evaluation.

Table 24: Functional requirements for the SPRINT Converter framework

ID	Description	Scenario	C-Rel status	F-Rel plan
CF-R1	The Converter framework must support creating batch converters.	D5.1 S5	OK	NO
CF-R2	The Converter framework must support creating service mediators.	D5.1 S6	OK	YES
CF-R3	The Converter framework must be able to access, process and produce different data formats.	D5.1 S5, S6	OK	NO
CF-R4	The Converter framework must support the definition of conversion pipelines through a configuration file.	D5.1 S8	OK	NO
CF-R5	The Converter framework must integrate the ST4RT annotations for lifting and lowering.	D4.1	OK	NO
CF-R6	The Converter framework must support lifting based on declarative mappings.	D4.1	OK	NO
CF-R7	The Converter framework must support lowering based on declarative mappings.	D4.1	OK	NO
CF-R8	The Converter framework must support the definition of preprocessing/custom steps in the conversion pipeline.	D4.1	OK	NO
CF-R9	The Converter framework must support data enrichment considering external data sources during the conversion.	D4.1	OK	NO
CF-R10	The Converter framework must support inference enrichment considering a set of ontologies.	D4.1	OK	NO
CF-R11	The Converter framework must be configurable to run as a standalone jar.	D5.1 S5, S8	OK	NO
CF-R12	The Converter framework must support the definition of endpoints to request a runtime conversion.	D5.1 S6	OK	NO
CF-R13	The Converter framework must be deployable as a Docker image.	D4.2, D5.1 S8, S9	OK	YES

6.3 PERFORMANCE AND SCALABILITY REQUIREMENTS

This section reports the performance and scalability requirements, described in D3.2, and provides a summary about which requirements are already fulfilled in C-Rel and which requirements will be further investigated during F-Rel.

6.3.1 Asset Manager

Table 25: Asset Manager performance and scalability requirements

Description	Ref. the to full description of the Requirement and KPIs	C-Rel status	F-Rel plan
Performance requirements for Ad-hoc Asset creation: (exemplified scenario for Converter asset)	Requirement in Deliverable **D.3.2**: Table 13 – Req 2	OK	YES
Scalability requirements for Direct Download of an Asset (exemplified scenario for Converter asset)	Requirement in Deliverable **D.3.2**: Table 16	OK	NO

6.3.2 Distributed SPARQL endpoint

Table 26: Distributed SPARQL endpoint performance and scalability requirements

Description	Ref. the to full description of the Requirement and KPIs	C-Rel status	F-Rel plan
Asset/Service Discovery Response Time	Requirement in Deliverable **D.3.2**: Table 13 – Req 1	OK	NO
Scalability of Query/search time	Requirement in Deliverable **D.3.2**: Table 14	OK	NO

6.3.3 Converter

The target values defined for the performance requirements of the IF Converter have been fulfilled. For F-Rel, we plan to address the issues related to the conversion of large XML files and to optimize further the IF Converter performances as discussed in Section 5.2.2.

The defined scalability KPIs for the IF Converter mainly depend on the adopted deployment. For F-Rel we will report results obtained in the final testing environment.

Table 27: Converter performance and scalability requirements

Description	Ref. the to full description of the Requirement and KPIs	C-Rel status	F-Rel plan
Response Time to convert the whole data set	Requirement in Deliverable **D.3.2**: Table 15 – Req 1	OK (issue with XML datasets)	YES
Response Time to convert one message	Requirement in Deliverable **D.3.2**: Table 15 – Req 2	OK	YES
Scalability requirements for Runtime environment deployment of an Asset (exemplified scenario for Converter asset)	Requirement in Deliverable **D.3.2**: Table 17	NO	YES

6.3.4 Mapping suggestion tool

Table 28: Mapping suggestion tool performance and scalability requirements

Description	Ref. the to full description of the Requirement and KPIs	C-Rel status	F-Rel plan
Execution Time	Requirement in Deliverable **D.3.2**: Table 15 – Req 3	OK	YES
Usability	Requirement in Deliverable **D.3.2**: Table 15 – Req 4	OK	YES

7. REFERENCES

[1] SPRINT, "D3.2 - Performance and Scalability Requirements for the IF," 2019. [Online]. Available: http://sprint-transport.eu/Page.aspx?CAT=DELIVERABLES&IdPage=1e2645be-e780-4d99-8117-bae57b67b453.

[2] SPRINT, "D3.3 - Design of Architecture, Testing Infrastructure, Test Cases and Benchmarks of the IF (C-REL)," 2019. [Online]. Available: http://sprint-transport.eu/Page.aspx?CAT=DELIVERABLES&IdPage=1e2645be-e780-4d99-8117-bae57b67b453.

[3] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini and R. Rosati, "Linking data to ontologies," *Journal on data semantics X,* vol. 10, pp. 133-173, 2008.

[4] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro and G. Xiao, "Ontop: Answering SPARQL queries over relational databases," *Semantic Web,* vol. 8, 2016.

[5] D. Chaves-Fraga, F. Priyatna, A. Cimmino, J. Toledo, E. Ruckhaus and O. Corcho, "GTFS-Madrid-Bench: A Benchmark for Virtual Knowledge Graph Access in the Transport Domain".

[6] "GTFS Static Overview," [Online]. Available: https://developers.google.com/transit/gtfs/.

[7] D. Lanti, G. Xiao and D. Calvanese, "VIG: Data scaling for OBDA benchmarks," *Semantic Web,* vol. 10, no. 2, pp. 413-433, 2019.

[8] A. Dimou, M. V. Sande, P. Colpaert, E. Mannens and R. V. d. Walle, "Extending R2RML to a Source-independent Mapping Language for RDF," in *International Semantic Web Conference (Posters & Demos)*, 2013.

[9] S. Das, S. Sundara and R. Cygania, "R2RML: RDB to RDF Mapping Language. W3C Recommendation," September 2012. [Online]. Available: https://www.w3.org/TR/r2rml/.

[10] F. Michel, L. Djimenou, C. Faron-Zucker and J. Montagnat, "xR2RML: Relational and Non-Relational Databases to RDF Mapping Language," 2014.

[11] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro and G. Xiao, "Ontop: Answering SPARQL queries over relational databases," *Semantic Web,* vol. 8, no. 3, pp. 471-487, 2017.

[12] SPRINT, "D5.2 – Software release of the proof-of-concept in its technical environment (C-REL)," 2020. [Online]. Available: http://sprint-transport.eu/Page.aspx?CAT=DELIVERABLES&IdPage=1e2645be-e780-4d99-8117-bae57b67b453.

[13] ST4RT, "D5.1 - Requirements for the Demo," 2018. [Online]. Available: ttp://www.st4rt.eu/Page.aspx?CAT=DELIVERABLES&IdPage=3cd536b1-4a34-4432-8987-115810d01dd3.

[14] ST4RT, "D5.2 - KPI and metrics for evaluation," 2018. [Online]. Available: http://www.st4rt.eu/Page.aspx?CAT=DELIVERABLES&IdPage=3cd536b1-4a34-4432-8987-115810d01dd3.

[15] ST4RT, "D5.4 - Report on the Results of the Pilot," 2018. [Online]. Available: http://www.st4rt.eu/Page.aspx?CAT=DELIVERABLES&IdPage=3cd536b1-4a34-4432-8987-115810d01dd3.

[16] S. Borzsony, D. Kossmann and K. Stocker, "The skyline operator," in *Proceedings 17th international conference on data engineering*, 2001.

[17] S. Jozashoori and M. E. Vidal, "MapSDI: A Scaled-Up Semantic Data Integration Framework for Knowledge Graph Creation," in *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems*, 2019.

[18] SPRINT PROJECT, "D2.2 REQUIREMENTS FOR AN IF ARCHITECTURAL DESIGN (C-REL)," 2019. [Online]. Available: http://sprint-transport.eu/download.aspx?id=7b3a7122-16f5-43e6-9a4f-f749c24e1592.

[19] ST4RT Consortium, "www.st4rt.eu," [Online].

[20] SPRINT, "D5.2 - Software Release of the Proof-Of-Concept in its Technical Environment (C-REL)," 2019. [Online]. Available: http://sprint-transport.eu/Page.aspx?CAT=DELIVERABLES&IdPage=1e2645be-e780-4d99-8117-bae57b67b453.

[21] M. Glinz, "On non-functional requirements," in *n 15th IEEE International Requirements Engineering Conference (RE 2007)*, 2007.

[22] Semantic Web Group, "W3C SEMANTIC WEB ACTIVITY," 11 December 2011. [Online]. Available: https://www.w3.org/2001/sw/. [Accessed 14 August 2019].

[23] D. Chen, G. Doumeingts and V. François, "Architectures for enterprise integration and interoperability: Past, present and future," *Computers in industry 59(7),* pp. 647-659, 2008.

[24] M. Glinz, "On non-functional requirements," in *In 15th IEEE International Requirements Engineering Conference*, 2007, October.

[25] I. Jacobson, The unified software development process, Pearson Education India, 1999.

[26] H. H. Liu, Software performance and scalability: a quantitative approach, John Wiley & Sons, 2011.