Contract No. H2020 – 826172

SEMANTICS FOR PERFORMANT AND SCALABLE INTEROPERABILITY OF MULTIMODAL TRANSPORT

D2.5 Recommendation to IP4 – final version

Due date of deliverable: 31/12/2020

Actual submission date: 01/04/2021

Leader/Responsible of this Deliverable: UITP

Reviewed: Y

Document status		
Revision	**Date**	**Description**
0.1	07/10/2020	Table of Content
1.0	15/12/2020	First version
1.1	25/01/2021	Final version for TMC approval
1.2	28/01/2021	Final version submitted after TMC approval
2.1	09/03/2021	Revised version following comments of the JU
2.2	29/03/2021	Final version of revised deliverable for TMC approval
2.3	01/04/2021	Final version of the revised deliverable submitted 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: 27 months

EXECUTIVE SUMMARY

This report is the final output of SPRINT Task 2.4 concerning the recommendation to IP4 after the validation of the F-REL proofs-of-concept in WP5. This report is the **updated version of D2.4** as the results of the validation of the F-REL proofs-of-concept didn't change dramatically the outcomes of C-REL. Only new insights after F-REL validation were added in the current version of the recommendations. The work includes outputs of WP2, WP3 and WP4, and provide the final recommendations on future developments and deployment of the IF to S2R IP4.

Section 2 delivers recommendations based on analysis of requirements of S2R IP4 projects and related EU initiatives. This section was not updated since the issue of D2.4. The recommendations still stay actual, and they are based on the outcomes of D2.1. Section 3 is focused on recommendations for the IF architectural design. Recommendations for the IF architecture as a component to NAP are presented in Section 4. Section 5 provides recommendations for the performance and scalability of the IF, including lessons learned from F-REL implementation and validations. Recommendations for the IP4 IF semantic automation are delivered in Section 6. Finally, recommendations related to the market uptake can be found in Section 7.

ABBREVIATIONS AND ACRONYMS

Abbreviation	Description
API	Application Programming Interface
BPMN	Business Process Model and Notation
CMMP	Contractual Management Market Place
EIF	European Interoperability Framework
FSM	Full Service Model
GOF4R	Governance of the IF for Rail and Intermodal Mobility
IF	Interoperability Framework
IP4	Innovation Programme 4
KPI	Key Performance Indicator
MaaS	Mobility As a Service
NAP	National Access Point
RDF	Resource Description Framework
RML	Relational Meta Language
S2R	Shift2Rail
SHACL	Shapes Constraint Language
ST4RT	Semantic Transformations for Rail Transportation
TSP	Transport Service Provider
UML	Unified Modeling Language
XML	Extensible Markup Language
W3C	World Wide Web Consortium
WP	Work Package

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

This report is the final output of SPRINT Task 2.4 concerning the recommendation to IP4 after the validation of the F-REL proofs-of-concept in WP5. This report is the updated version of D2.4 as the results of the validation of the F-REL proofs-of-concept didn't change the outcomes of C-REL dramatically. It includes outputs of WP2, WP3 and WP4, and provide the final recommendations on future developments and deployment of the IF to S2R IP4.

The report defines recommendations for IP4 to support the market uptake by

(i) Providing solutions satisfying requirements of both IP4 members and external EU initiatives.

(ii) Simplifying/automating all the necessary steps which are needed to integrate new services and sub-systems in the multi-modal transport ecosystem.

(iii) Emphasizing the potential role of the IF for NAPs.

(iv) Taking care of improving performance and scalability of the IF following its development and deployment.

(v) Showing how the IF contributed to addressing different market uptake indicators.

2. RECOMMENDATIONS BASED ON ANALYSIS OF REQUIREMENTS OF S2R IP4 PROJECTS AND RELATED EU INITIATIVES

Based on the analysis of requirements of S2R IP4 projects (CONNECTIVE and ATTRACkTIVE) and different EU initiatives in SPRINT D2.1, the following recommendations to the IF can be proposed:

- **Leveraging automation**

In the future, services developed in the IP4 projects or by the TSPs involved in the IF should be not annotated with meta-data that facilitates their discovery. Since the ecosystem is now rather small, there is not a specific need for automated service discovery. However, as the ecosystem grows, service discovery is expected to become an issue.

Furthermore, the creation of services is done manually for now. In future, it is important to support techniques that would allow service providers – for example, TSPs – to specify the configuration of their services, which would then be traduced into services (or skeletons thereof).

As the discovery of external services is done manually, the process of its automation should be also considered in the following versions of the IF.

- **Monitoring and Governing**

While the IF ecosystem is rather small and limited to the members of the IF4, there is no issue of monitoring and governing the assets. However, with the deployment and scalability, the governing rules should be very well defined and followed by the ecosystem members. More specifically:

- Levels of access to services and assets must be defined (for example, to limit the types of users that could access certain data, for example, the so-called meta-network of a TSP). If any restrictions are necessary, in the future they should be regulated through legal contracts (for instance, integration with CMMP).

- Workflow management and version handling. Artefacts handled in the IF ecosystem (e.g., provided services and data) are not managed now through a codified lifecycle for their creation/update/destruction. So, with the scalability of the IF ecosystem, the workflow has to be set up.

Analysis of initiatives for different modes of transport, including the new ones (e.g., combined mobility operators, MaaS operators) have different requirements to the interoperability, so to consider these requirements in the following versions of the IF ecosystem, representatives of these modes should be included in trials and testing. To put all types of stakeholders together require a specific governance body (e.g., ITxPT, MASAI), described in GOF4R D5.1 – Deployment Roadmap.

- **Leveraging the technological neutrality**

The IF development and deployment should follow the principle of technological neutrality it will help the technology to be pushed to the market and find the right structure interoperable for different stakeholders, even outside the rail sector.

The EIF provides a range of recommendations to the IF which can be considered during the IP4 IF deployment and scalability process.

Collaboration with ERTICO to set up common interoperability rules can be launched. ERTICO supports DATEX II model, which is a standard model at the European level for the exchange of data related to traffic. This standard does not support semantic interoperability but a possibility could be to study if the transformation of DATEX II to an ontology would improve the management and interoperability among the used datasets, as it has identified in the public transport where efforts in the transformation of NeTEx to a corresponding ontology have started.

Deployment of the IF for NAPs is another challenge to consider by S2R. An overview of the NAPs across Europe shows that the NAPs vary in system architecture, organisation, monitoring of data users, accessibility, etc. Thus, there is a need for a more coordinated approach and exchange of ideas and best practices. Theoretically, the IF ecosystem can become such a solution, so the involvement of different Member States is required.

3. RECOMMENDATIONS FOR THE IF ARCHITECTURAL DESIGN

In this section, we first overview the most important lessons learned during the IF architecture requirement analysis as well as during the F-REL development of the IF, along with general discussions and recommendations. Then, the concrete and specific recommendations for further development of the IF architecture are summarized at the end of the section.

Lessons learned and related discussions and recommendations

The most prominent lesson we have learned within this project *was* ***to avoid the centralized deployment of the IF***, since it may jeopardize its overall scalability and robustness. Having one IF node (a software unit running in one server) as the single (physical) point of interaction that is responsible for the coordination and running of the whole aspects of IF leads to the single-point-of-failure problem. More importantly, it would become a performance bottleneck and it would prevent the scaling up as the eco-system grows.

The SPRINT project, in particular, has stressed all along that the concept of the IF as a monolithic software with centralized deployment, offering a fixed set of services/components which are over-tailored to work with certain transportation services is not suitable.

Firstly, the IF should not be perceived as a middleware that itself mediates the non-interoperable interactions between different parties. Rather, it is an infrastructure that offers base services, components and utilities (such as Converter, Automated Mapping, Semantic-Based Discovery. etc.) as the enablers of interoperability to the interested transportation actors. Hence, there is a need for having a federated IF, that is multiple IF Nodes distributed across Europe. While IF nodes could be in communication with each other, their operations are stand-alone and tailored to a particular region.

In this direction, our recommendations are as follows:

- The IF should be implemented in a distributed manner, without having a single IF Node responsible for the whole European Union. In particular, we recommend one IF node per National Access Points, but the distribution and granularity of regions could be possibly any of the following:
  - *one IF Node per EU country* (which occurs if each IF node acts as National Access Point), or
  - *one IF Node per district*, or
  - *one IF Node per major transport operator* (e.g., SNCF, Trenitalia).
- Yet, the complexity of the distributed system must be hidden from users. Hence, we recommend providing users with a single point of interaction to have access to the various IF functions. In particular, this could be achieved through the utilization of design patterns such as API Gateway [1] that leads all users to a single (logical) access point where the gateway redirects the requests to the desired IF instance/server.

The latest implementation of IF is aligned with abovesaid design recommendations. The focus of the SPRINT project was to realize the first instance of IF, rather than a distributed network of IF nodes. However, it has been designed and implemented having the vision of a distributed deployment.

Hence, the federation of IF could be considered as the next step of IF ecosystem development and the extension to the SPRINT IF.

Secondly, in the SPRINT project, we believed that interoperability should be established not only in the components and services offered by the IF but also in its architecture and the way it has been built and deployed. Accordingly, the IF in the SPRINT project has been designed and developed as a **modular set of self-contained components** that can be offered in an IF Node.

In this direction, to maximize the extendibility of IF, and to enhance engagement with the IF, it should offer its functions/services in a manner that is as fine-grained as possible. This is achieved by following a decomposition strategy based on the business purpose, requirements, and function, but hiding such complexity from users. In particular, we recommend the following:

- A microservice-based architecture, rather than a monolithic architecture.
- The IF should let users selectively opt for the various functions based on their needs and use only those components of the IF in which they are interested.
- The IF should allow interested parties to extend any desired services and components independent from the rest of the components and services.

The latest development of the IF developed in the SPRINT project is already in line with the above recommendations, both for what concerns the architecture of the IF itself, as well as the architecture of its internal components. As demonstrated in Deliverable D5.3, the components of the IF are self-contained, they can be deployed in a stand-alone manner, and they can be registered and later discovered to/via Asset Managers. Furthermore, where modularization is applicable, even single components of the IF can be created by the composition of multiple modules. For example, in scenario 8 in D5.3, users can customize the inner modules of a Converter and build a customized Converter based on their needs.

Another important observation we made was a subtle mistake that may occur for the development of IF, that is to consider it as a giant shared data centre for all the partners to store and share their data. The objectives of the IF, however, go **beyond the provision of a data-sharing framework**. The IF, instead, aims to facilitate technological and data interoperability that lets organizations interactively cooperate to use each other's data and services as seamlessly as possible and to build new services and utilities. Our strong recommendation in this regard is

- To avoid storing data in the IF and reducing its function to a data storage/sharing infrastructure.

Regarding data sharing, another essential consideration is that **data ownership always matters.** The willingness of business parties and organizations to expose their data usually comes with a strong desire to keep ownership and full control of their data. The IF architecture and associated technologies, then, must comply with such demand by design and let the owners fully control the access rights to their data. In the current implementation of the IF, the owner of data and assets can control who can access the published assets.

- In line with the distributed nature of the IF architecture, as well as the importance of preserving the full control of owners over the access policy of their data, we recommend that the accessibility to the system should not be governed by a centralized authentication and authorization system. A distributed access control mechanism is recommended that lets individual business parties and organizations have full control over their authorization policies.

Hence, to avoid shaping the IF as a data storage centre, and to fully guarantee data ownership, the Asset Manager of the IF has not been designed as a shared database for contributors to upload their assets. Instead, it is a catalogue for assets that contains only meta-data and asset descriptions. Given this design, then, the Asset Manager mainly keeps and advertises the "meta-data" of assets and not the assets themselves which allows the data owner to preserve full ownership of such assets. Furthermore, to have a flexible privacy and access control management for the owner of the assets, the concept of User Manager has been introduced in the IF. It lets the owner of assets configure the accessibility and visibility of their assets and fully control who can see/have access to what. (Please refer to Deliverable D.5.5, for a full description of the F-REL implementation of Asset Manager and User Manager).

Moving to another dimension, an important lesson we learned during the SPRINT project, and, in particular, in our collaboration with the CONNECTIVE project, was that **the integration with systems currently in use matters.** The goal of the IF should not be to replace existing functions, and it should not create any process duplications; rather, it should re-use the current infrastructure and integrate with existing systems. For example, the Operator Portal is an existing and well-established framework currently to facilitate user management in the transportation domain. The IF hence can integrate with such a system to let users seamlessly log-in to the IF instead of creating yet another registration portal and mandating users to re-do the registration process over and over again. Furthermore, such integration greatly decreases the administrative burden on the users. In particular, we recommend the following:

- Facilitate registering to and joining the ecosystem by providing a single-sign-on solution. This also makes it simpler for various transportation-related organizations and operators to (logically) enable their users to use the IF easily and seamlessly.
- Leverage the distributed and cross-organization collaboration.

In line with these considerations, the current Asset Manager is integrated with an Identity Provider to provide maximum interoperability with other systems such as the Operator Portal. Hence, users can register to either the Asset Manager or the Operator Portal and have access to both systems.

Another critical insight we gained starting with the requirement analysis, but also later during the implementation and the development of the IF was that the **IF should offer options**. Different users with different business goals and demands and different level of technical expertise need to engage with the IF differently. Hence, the IF should avoid constraining users to a single option for engaging with the IF and its component, but it should foresee multiple possibilities and solutions. In particular, we recommend the following:

- The IF should provide different deployment options, from the direct download of components to a service-based model, for different categories of users and business partners.
- The IF should try to minimize any technical requirements and reduce the barriers of entering the ecosystem and facilitate engagement with the IF for different categories of users and business partners.

This consideration has been at the centre of our design goals and the current IF implementation enables various engagement and deployment possibilities. Such options have been successfully developed and validated in Deliverable D5.3, and specifically through scenarios S4, S5 and S6.

Another important message we would like to highlight here is that **Automation Matters**. Indeed, automation plays a pivotal role to increase interoperability. It breaks a complex process into intermediate steps and provides a formal description for each step to be processable by machine. Accordingly, the existence of such a formal and machine-understandable description of procedures further leads to the possibility of formalizing the interoperability and integration without any human intervention. In this direction the IF architecture should promptly foster automation, in particular, we recommend the following:

- The IF should support automatic software building, in particular through the realization of continuous integration/delivery tools.
- The IF should support automatic/semi-automatic deployment via deployment scripts.

The current implementation of the IF has already taken such considerations at heart and it contributes to this vision in many ways. For example, the Mapping Tool is one of the components of the IF that particularly offers a semi-automated mapping generation between heterogeneous standards. The final validation of this tool in Deliverable D5.6 showed promising results. Furthermore, the IF enables many automated procedures for the creation as well as the deployment of Converters, as shown in scenarios S8 and S9 of D5.6. Finally, the IF offers various contributions on Semantic Automation and the corresponding recommendations are reported in this document in Section 6.

Last but not least, it is evident that semantic web technologies win. Semantic technologies greatly help to overcome existing heterogeneity and lead to “a common framework that allows data to be shared and reused across application, enterprise, and community boundaries” [1]. In this direction the IF architecture must natively support semantic technologies, so we recommend the following:

- The incorporation of semantic meta-data and descriptions into asset descriptions.
- To favour semantic-based searching and discovery, in particular through distributed SPARQL endpoints.

Semantic technologies are one of the pillars of IF. In a nutshell, the IF is designed to both use semantic technologies to enhance semantic interoperability – for example through distributed SPARQL endpoints, as well as meta-data annotations of the assets – and to provide enablers to foster semantic technologies and approaches – for example through ontology management utilities. The F-REL implementation and validation of all the above tools in deliverable D.5.6 have shown promising results for achieving the above-mentioned goals.

Summary of recommendations:

- Avoid implementing the IF as a centralized framework as well as a centralized data centre;
- Distribute the instances of IF nodes within European Union regions, possibly having one per National Access Points;
- Avoid a monolithic architecture;
- Favour the use of structured formats, such as ontologies and vocabularies, to describe data, to provide structured and machine-readable service descriptions, etc;
- Favour the use of semantic discovery, and in particular of distributed SPARQL endpoints, to provide unified access to a complementary set of (sometimes overlapping) knowledge graphs;
- Architect the IF as a modular software and in compliance with the Service-Oriented Architecture paradigm. In particular, we recommend implementing the IF and its components as microservices;
- Favour the API Gateway pattern for the microservice-based architecture;
- The IF should store only meta-data and not concrete data;
- The IF should provide a registry that lets organizations advertise data by sharing the “meta-data” only;
- Use of any data (if they are not open) should be completely controlled by the owners of data;
- Integrate with Operator Portal to facilitate the single-sign-on process;
- Minimize the set of services/component for instantiating IF;
- Provide the possibility of extending IF features, functions and components upon the needs of participating actors;
- Ensure deployability of IF services, in particular by the deployment of IF components/services through container technology that packages such components as portable, self-contained and ready-to-run software units;
- Emphasize a plug-and-play approach and provide means for the automated generation of software units.

4. RECOMMENDATIONS FOR THE IF ARCHITECTURE AS A COMPONENT TO NAP

D2.3 defines different usage scenarios about how the Interoperability Framework can relate to National Access Points. As we described in the document, the IF can be used as a way to implement a NAP, or as a companion to existing NAPs. The features which are deemed as mandatory and nice to have are all covered by the current version of the SPRINT tools, which provide features far more advanced than what the current National Access Points implementations provide. The integration between the Asset Manager and the Converters allows automatic dataset conversion, and therefore an IF-based NAP could hide the complexity of Transmodel-based standards to all TSPs currently providing their data in different formats (like GTFS). The possibility to define complex lifecycle management processes moreover allows for finer-grained and automated control of the process and its implied roles and responsibilities.

Though the IF (as described in the various SPRINT deliverables) can play the role of a National Access Point, each member state is already designing or providing a solution that is not based on the IF. Therefore, a realistic role for the IF is as a companion to NAPs, to ease obtaining metadata from multiple sources and to contribute to a specific NAP according to the regulations. While implementing and testing our prototype we identified several key elements to be taken into account to integrate the Interoperability Framework and the National Access Points.

In the context of F-REL, we successfully demonstrated how the Asset Manager can become aware of assets published in National Access Points, and we identified human-based methods to try to deal with metadata quality issues. In the following subsections, we will describe how we investigated such elements, what we discovered and which suggestions can be proposed to Shift2Rail IP4.

4.1 METADATA INTEROPERABILITY

While the role of multimodal National Access Points and their data compliance rules are clearly stated inside the EU regulations, no specifications are mandated regarding how each EU member state should implement such National Access Points. Therefore each NAP is implemented with different technology, has different API features, and a different metadata set. In D2.3 we listed and analysed some cases, finding that some countries are using Open Data Portals like CKAN to implement their NAP (like Belgium), while others are exposing SPARQL endpoints (like the Czech Republic and Denmark). In SPRINT F-Rel we exploited a joint work made by Germany, The Netherlands and Austria (Coordinated metadata catalogue) aiming for common metadata set for all the European National Access Points. We also demonstrated that it is possible to map different metadata schemas onto DCAT-AP 2.0.1 schema and that such mapping enables the possibility to use the Asset Manager as an aggregator of several NAPs.

By using the Asset Manager as an aggregator, the Interoperability Framework could benefit from a side-effect: any asset published in a NAP comes from a trusted source, and any TSP who publishes an asset in a NAP must have previously signed a “contract” which guarantees data quality. This means that a NAP-enabled IF would require fewer interactions from the TSP side in order to obtain relevant data. In a future scenario where NAPs are fully running and where timetables and real-time information can be automatically located on NAPs and fetched, there would be no need to ask again the same datasets to TSPs joining the IF. After successful registration, the Asset Manager could simply fetch relevant assets from NAPs, and then ask the TSP whether he wants to re-use the same

information in the IF ecosystem. That would help reduce information duplication and would minimize errors due to data misalignment between NAPs and IF.

4.2 DATA COMPLIANCE

The first data constraint imposed by the NAP is the usage of Transmodel-based specifications. NeTEx, DATEX, SIRI, are all standards and specifications derived from Transmodel, which acts as a common conceptual model. The first recommendation is, therefore, to align the IT2Rail/IP4 ontology to Transmodel (which should be turned into an ontology itself), and to provide a mapping to be able to import data from NAPs into the IP4 ecosystem and to export data complying to the regulations. An important requirement of this alignment is that it must be “lossless”, i.e. it must be possible to import all the information contained in a NeTEx dataset, and to export all the information to that format.

Another option for such alignment, although with a higher cost, is to re-design the IT2Rail/IP4 ontology as an extension to Transmodel. That would require re-writing the Service Implementation inside Brokers to reflect the model changes but would widen the adoption of the IF since Transmodel being pushed as a central element of NAPs means it will become a sort of “official” model for all EU transport operators.

4.3 ACCESSING AND CONTRIBUTING DATASETS

D2.3 defined two different roles for the interaction between the IF and a NAP. The Asset Manager, playing the role of the “catalogue of the IF”, can be an aggregator of metadata coming from several NAPs, allowing TSPs to acknowledge the existence of relevant datasets shared across Europe. The Asset Manager, being also a publication and sharing platform, can also help TSPs being compliant with the EU regulations and publish datasets to National Access Points on behalf of a TSP.

In our F-REL prototype, we tried to implement a demo of both usages of the Asset Manager. We managed to aggregate metadata coming from three different National Access Points (France, Netherlands and Belgium), and we discovered a huge metadata quality problem. The Asset Manager allows defining several “asset types” (like ontologies, timetable datasets, mappings), and while dealing with NAPs the first issue to solve is how to categorise remote assets into locally-defined asset types. The EU regulations mandate that NAPs must become repositories of transport datasets, but they don’t specify how a member State should implement its solution. We realized that a common trend is to implement NAPs as a part of the open data effort, and therefore transport data is simply added to open data portals already in place. From a technical point of view, this means that if we’re trying to access a NAP looking for datasets to be locally categorised as “timetable datasets”, we can only search using a generic “transport dataset” label, and the results can vary from actual timetables to road traffic condition or weather reports on roads. This is due to the fact that there is no fine-grained categorization on currently available NAPs. To become useful in an automated ecosystem, the IF (via the Asset Manager) should define a two-step metadata acquisition process. During the first step (which we already implemented), the metadata are downloaded from remote NAPs and are harmonized using DCAT-AP 2.0.1. During the second stage, such “candidate assets” are manually revised and become part of the catalogue only after explicit approval. Such approval could be granted by the IF administrators or even by TSPs who previously published their datasets to their National Access Point.

While aggregating metadata requires addressing metadata quality issues, contributing to a NAP via the IF requires dealing with delegation. Each NAP implements a different procedure to register a new TSP and to contribute new datasets, and not all of them are compatible with the concept of an "aggregator". One of such scenarios is represented by contributions enabled by a security certificate issued by the NAP. In such cases, the contributor must connect to the NAP using the certificate, which is strictly personal and should not be distributed to any third party. This means that a "NAP-enabled Interoperability Framework" can easily benefit from data being published inside NAPs and that directly contributing to existing NAPs depends on the deployment model for the IF. If the IF will be deployed as a federation of "interoperability nodes" each one owned by a different TSP, it will become possible for a TSP to add custom publication rules including contributing to NAPs. If the IF will be deployed as a central "hub", then delegating contributions to NAPs to the IF may become technically impossible. In such a case the IF could keep its role of a "read-only" NAP companion, and could anyway leverage on the availability of officially approved transport data.

Summary of recommendations:

- Define a Transmodel ontology;
- Align the IT2Rail/Shift2Rail ontology to Transmodel or rewrite it as an extension to Transmodel;
- (In case the two ontology are simply aligned) define mappings to convert data between Transmodel and IT2Rail/Shift2Rail data models;
- Define a common metadata ontology as a superset of the existing NAP metadata schemas, following the scheme used to implement our NAP aggregation scenario;
- Develop Converters to import metadata from NAPs according to the to-be-defined metadata ontology;
- Develop Converters to export metadata to the specific format adopted by the destination NAPs;
- Implement NAP-aware publication processes for selected asset types (like Journey planning);
- Implement a metadata quality assurance process to allow users to access NAP assets only if their metadata quality have been explicitly approved by the administration staff.

5. RECOMMENDATIONS FOR PERFORMANCE AND SCALABILITY OF THE IF

First and foremost, a significant point that must be highlighted here is that the scalability and performance of the IF can be considered and analyzed from two different perspectives: first, for the IF as a whole and, second, for its components. In the following, we first focus on the scalability and performance considerations for the whole IF, on the related challenges and the corresponding recommendations. Then, we discuss the performance and scalability issues of individual components of the IF, how they can be scaled and achieve performance targets, what are the related challenges and bottlenecks and the corresponding recommendations, mainly based on the results and analysis of the C-REL and F-REL implementation and of the validation of such components.

Scalability and performance are categorized as two different properties of a software system, but they are highly correlated. For a given environment that consists of properly-sized hardware, properly-configured operating system, and related middleware, if the performance of a software system deteriorates rapidly with an increasing load (number of users or volume of transactions) before reaching the intended load level, then it is not scalable and it will eventually underperform [1].

In this regard, the deployment of strategy the IF – i.e., centralized vs distributed – becomes the key factor in managing the performance and scalability of the IF. Accordingly, we recommend practising the federated deployment of multiple IF instances distributing the load throughout many nodes that are cooperating to create a holistic distributed infrastructure. This approach – in comparison with the centralized model where one single node is responsible to manage every aspect and ever-increasing user's loads – enhances the overall performance of the system and ensures the scalability of the IF to become a framework used by considerably large numbers of transportation operators and actors all over Europe.

Accordingly, enhancement of the scalability and performance of IF is mainly reflected in the architecture of the IF and the architectural design decisions are the key factor to balance the scalability and performance requirements of the IF. Hence, our recommendations concerning the scalability and performance of the IF are in line with those mentioned in Section 3, and they are as follows:

- Avoid a centralized implementation of the IF;
- A federated IF is recommended, in particular as a materialization of NAPs;
- The IF should not be used as data storage;
- The IF should only store meta-data;
- The initial setup of the IF must be minimized so users can add only those components and uses of the IF according to their needs and performance constraints.

Such recommendations refer to the IF as a whole, but additional recommendations can be defined for specific components:

- User manager: carefully decide whether to centralize authorization rules or to use such component for authentication alone, implementing authorization in each single other IF component. Both approaches can be valid and present advantages and disadvantages, both in terms of scalability and required management effort.
- Asset Manager: Exploration APIs can provide a way to implement Resolvers based on the RDF metadata contained in the catalogue. Moreover, the same mechanism of "parametric

SPARQL queries" can be implemented independently from the Asset Manager to speed up the implementation of RDF-based API.

- Asset Manager: the experience of integrating external metadata sources (the National Access Points) proved that it is possible to use this IF component as an aggregator of several repositories of trusted data and metadata.
- Converter: our Chimera framework proved to be ready for adoption in a wide number of cases, both for the conversion of datasets in batch mode and for the service mediation case.
- Converter: the declarative approach to conversion using RML for lifting and Apache Velocity templates embedding SPARQL queries proved to be ready for dealing with both high conversion frequency and dataset conversion. The annotation-based approach already used in ST4RT and ported to our new framework proved to be useful in the case of service mediation, where the size of messages is not big, and developers are accustomed to relying upon marshalling and unmarshalling Java objects in memory.
- Distributed SPARQL Endpoint: the implementation was based on having a distributed SPARQL endpoint that runs queries on multiple data sources, and we can say that the Distributed SPARQL endpoint not fully support SPARQL 1.1 operators and produce an incorrect number of results obtained differs concerning the baseline.
- Distributed SPARQL Endpoint: the performance and scalability are very low when the data size increases, this is because the query federation tool does not have optimizations designed when the data source scales, and it is important to note there is still a lot of research about latency, optimizations and functions when executing a query on multiple sources.

In the next sections such recommendations, stemming from the F-Rel evaluation activities, will be thoroughly explained and motivated.

Also, a critical performance issue for the IF is the ability to handle a load of requests for the downloading of artifacts; this is yet another case that highlights the necessity of having multiple federated IF Nodes to scale up as the number of download request increases in such a way that the system can sustain its regular functionality without suffering a slowdown in its overall performance.

Furthermore, employing (and anticipating) the suitable *Deployment Approach*[1] for each service/component can practically deal with scalability and performance issues.

For example[2], for the use case of the batch data conversion process using a SPRINT Converter, and the automated learning of similarities among multiple standards through the SPRINT Mapping Tool, since the process is accomplished off-line and not in very frequent cycles, *The Direct Download of Deployable Component* approach seems the best option. Through that, the consumer downloads a deployable converter/mapping tool artefact (JAR, Docker image) to use it locally. Hence, the responsible entity to ensure the scalability of the converter/mapping tool is the service consumer. So, to ensure scalability and performance of the IF our recommendations include the following:

- Outsource the performance management to consumers by favouring the Direct Download Deployment strategy for those components of the IF which are to be used as self-contained modules.

[1] for more details regarding various deployment approaches please refer to SPRINT Deliverable D3.2.

[2] for more details please refer to SPRINT deliverables D5.1, D5.2.

The reason behind this recommendation is that a single instance of a stand-alone IF component, for example, a converter/mapping tool with a reasonable performance profile (e.g., few hours for batch conversion and few seconds for the mapping process) is enough for each service consumer and in the case of higher demand, the consumer could horizontally scale up its system by running multiple independent instances of the converter/mapping tool, which in turn is an external activity concerning the IF Node.

There are however a group of IF components/processes which inherently do not have strict performance requirements, but they may impose scalability challenges. To give an example, the process of joining the IF (registration, role assignment, etc.) must be done only once, and the information provided in this step seldom changes. Subsequently, users can tolerate a slower process without jeopardizing or losing interest in the framework. Similarly, the discovery process typically includes multiple executions of a simple search operation, each time adding filters to the previous attempt. In such cases, the main threat is the ability of the IF to be able to bear with the increasing number of users operating with the system simultaneously. In this regard our recommendation is as follows:

- Outsource the performance and scalability management to the service providers by favouring the Direct Access Deployment strategy (see Deliverable D3.2) for the interoperability services of IF.
- Scale-up IF capacities by favouring the Runtime Environment Deployment by automatic composition, deployment and replication of IF components and services on distributed nodes through cloud orchestrators such as Kubernetes.

Finally, other performance-critical aspects of the IF are mainly related to the functions of the IF that must deal with some sort of real-time data processing. For example, Runtime Message Conversion (see Deliverables D5.1 and D5.2), which aims at converting messages exchanged between two parties – i.e., converting the message represented in the sender standard to the standard understandable by the receiver – in real-time. For instance, when a shopping application tries to discover various itineraries offered by different and heterogeneous TSPs, a swift conversion process is required to proceed with the shopping procedure. In such cases a reasonable performance of this component of the IF is highly critical, otherwise, it would become the bottleneck for the whole process.

- Avoid monothetic and complex services that might consume huge memory and processing time and favour modular and micro-service-based architecture for each component and sub-system of the IF to distribute the loads.
- Favour horizontal scaling strategy and replication of the services/components.

All the above recommendations have been in the centre of the concluding design and development of the IF for its final release in the SPRINT project.

5.1 LESSON LEARNED FROM F-REL IMPLEMENTATION AND VALIDATIONS

In this section, we discuss the outcomes of D5.3 and D5.6 functional validations

5.1.1 Asset Manager

The implementation of the scenarios defined in D5.1 for C-Rel and D5.4 for F-Rel showed that the Asset Manager is able to play the role of the ecosystem catalogue inside the Interoperability Framework. The architecture and implementation choices showed the following advantages related to flexibility and scaling:

- multiple authentication mechanisms can be used, therefore the Asset Manager can benefit from a wide array of possible identity providers (Google, Facebook, Github, …);
- leveraging on containerization, there is a clear separation of concerns and each component is devoted to a specific task (CMS, CI/CD, process automation, caching, long-running tasks, …);
- components can scale independently;
- even if the Asset Manager implemented in SPRINT is a complete rewrite with respect to what was delivered in IT2Rail, the overall stability is improved. Such a result was achieved by higher reuse of production-ready open-source software.

Currently, the only bottleneck related to the scalability of specific components is represented by the Process automation component. A BPMN process engine relies on a database to keep track of the statuses of different process executions, and such a relational database is a bottleneck. This disadvantage is anyway mitigated by the fact that the process engine manages the assets lifecycles and the process by which users can obtain permission to access a specific asset. Such processes are not heavily stressed, since lifecycle management happens only during publication time, and being the Asset Manager mostly devoted to organizations and developers ensures that the number of requests for gaining access to a specific asset will not rise dramatically over time.

As explained in Section 4, in F-REL we extended the features of the Asset Manager to cover the possibility of aggregating metadata coming from different National Access Points. The implementation performs metadata alignment and consolidates the resulting RDF triples inside the RDF repository, as shown in Figure 1. Such an approach is fully generic and can be extended to any external metadata source by just providing a new RML mapping.

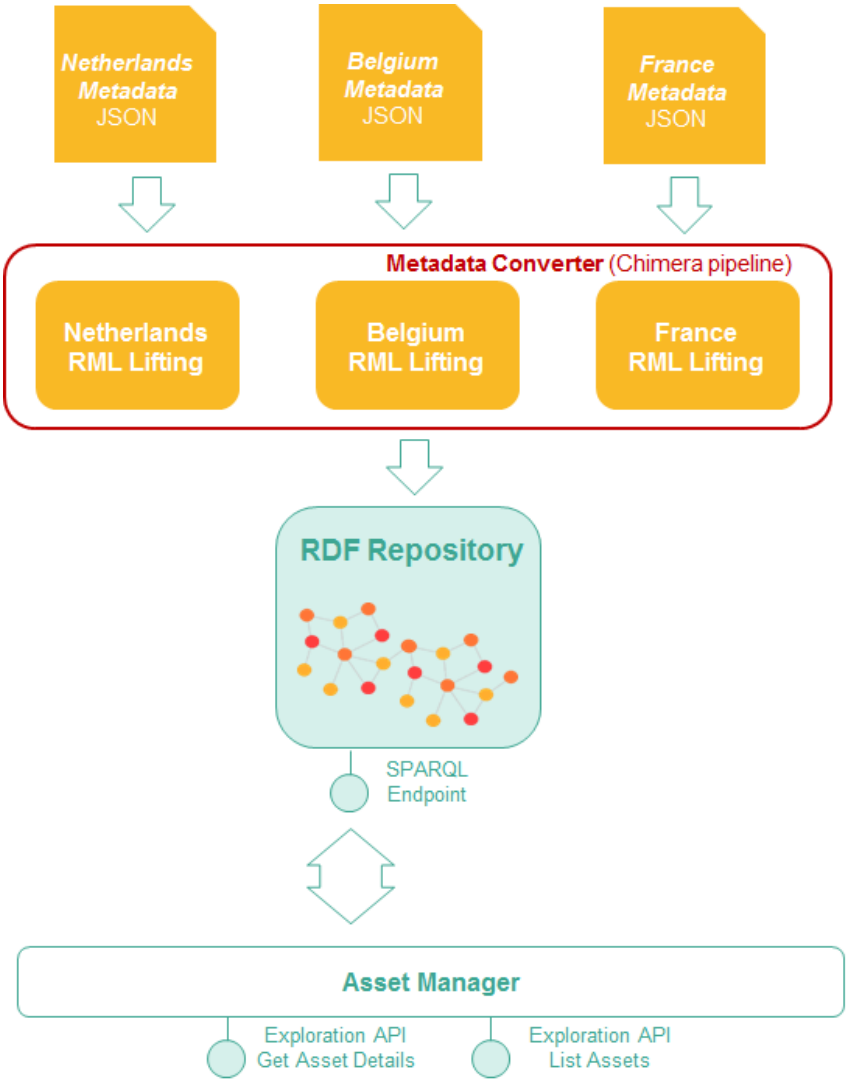

Figure 1: NAP metadata ingestion via Chimera and RML

The inclusion of NAP metadata in the RDF repository contributed to highlighting the importance of the Exploration API mechanism. Leveraging on such parametrized SPARQL queries exposed as Web API it was possible to develop nearly all the visible parts of the Web interface of the Asset Manager which needed interaction with the backend. Since an Exploration API is able to return JSON-LD data formatted via a JSON-LD Frame, such visual parts were developed with common Web libraries (such as Vue.js) without concerning with any Semantic Web-related issue.

The possibility to define a parametric SPARQL query and automatically expose an API to run it should be taken into account in the future developments of the IF, as it reduces the time required to implement a new API. Moreover, any software calling such API could either parse the results as a "normal" JSON object or consider them as RDF triples to be handled by Semantic Web-aware tools.

5.1.2 User Manager

The SPRINT IF architecture comprises a User Manager as the component devoted to authentication and authorization. In C-Rel such features are embedded inside the Asset Manager to speed up the

development process. The final version of the SPRINT IF successfully integrated the Identity Provider (IdP) deployed by the Connective project, and the Asset Manager can be configured to accept even more external providers. As we were pointing out in D2.4, the usage of widely accepted standards (like OAuth 2.0) allowed reusing open source libraries and offering extended single sign-on capabilities in a short time.

The integration with the Identity Provider deployed by Connective also showed the issue existing with the separation between authentication and authorization. To explain the issue, we must distinguish between Identity Providers owned by a third-party organization and Identity Providers owned by Shift2Rail IP4. In principle, all the IF components can rely on Identity Providers for authentication, the only condition being that such Identity Providers are somewhat considered "trusted" by the IF. Authentication relying on a long list of Identity Providers therefore can be a mean by which new users are "encouraged" to join the Shift2Rail IP4 ecosystem (since they can avoid registering on "yet another website"). Once registered, any user has the possibility to "to something" on the system. While in principle authentication can be delegated to external providers, authorization must instead be governed inside the IP4 ecosystem, because it deals with "who can do what" and this relates strictly with the components of the ecosystem itself. There are two ways to fulfil the requirement of managing authorization. One way is to add a "physically distributed but centrally governed" authorization layer, the other one is to let each ecosystem component define its roles and permissions. Obviously, each one of such ways has advantages and disadvantages. In the context of the collaboration between SPRINT and Connective, we tested the integration of the Identity Provider deployed by Connective and the Asset Manager provided by SPRINT. The Identity Provider in this case allows registering new users without dealing with authorization. This means that the Asset Manager had to define its internal roles and permissions and create an internal process by which a normal user can become a contributor to the IP4 ecosystem. As a result, the "IP4 Administrator" can know who is registered inside the ecosystem by looking at a single data source (the Identity Provider database), but he cannot know what a specific user can do inside the system without knowing which components are actually running and without asking each one of the components. In this scenario, strong collaboration among the owners of the systems/components composing the IP4 ecosystem must be put in place because there is no automated way to grasp what a specific user can do. Basically, we're trading flexibility (as each component owner can decide who can do what) for control (as it's difficult to fully understand what a specific user can do on which component).

The same Identity Provider (Keycloak) deployed by Connective could in principle be used to implement centralized authorization. With such an approach, the IdP is used for both authentication and authorization, and as such can be used to centrally govern "who can do what". Since it is the dual approach wrt. what we described before, it features a high level of control but it's not flexible. It would allow having a central user management console where administrators can treat the whole IP4 ecosystem as a single system, assigning roles and permissions to users being sure that such assignments are taken into account by all the components. Being centralized in a multi-lateral environment certainly diminishes flexibility, as this means that all parties must agree on the principle that the IP4 ecosystem must be treated as a single system. This implies the following list of required actions:

- List all the IP4 components
- List all the administrators of each component
- Obtain a list of the roles implemented by each component

- Obtain a list of permissions accepted by each component
- Create a centralized solution by which a user can ask permission to access a component or perform actions on a component

As can be clearly understood by the analyses of the two approaches for user management, it all depends upon the governance structure the IP4 ecosystem will use. If the Interoperability Framework and the higher-level components of the ecosystem will be considered as an open and federated environment, it is likely that light user management will take place, and each of the parties hosting components will manage users according to his own rules. If a legal entity will instead take care of deploying and maintaining the IP4 ecosystem, then centralized user management will be possible.

5.1.3 Mapping Tool

The final release of the Mapping Tool has a couple of enhanced features. Most importantly, while, the initial version of the Tool was a command-line application, the final version offers a Graphical User Interface to facilitate working with it and increase its user-friendliness. Furthermore, the automated generation of annotations, which are necessary for the conversion mechanism used by Converter components of the IF has been developed and integrated into the tool. The newly added annotations mechanism supports both Java-based annotations as well as RML-based annotation.

The mapping techniques also has gone through some improvements. The C-REL implementation was mainly based on the semantic similarity of the terms in the source and target standards but the F-REL has extended the work to also include Structural mapping to extract the similarity of the terms based on the syntactical structure of the source and target standards. With the inclusion of the structural mapping the overall accuracy of the algorithm has been improved as reported in deliverable D.5.6. Furthermore, the future work consists of the creation of a dedicated transport domain machine learning model which aims to cover specific vocabulary used in the transport domain. That will eventually improve the semantic accuracy of the mapping technique, and also it will provide a good ground for future research work in the transport domain.

5.1.4 Distributed SPARQL endpoint

In C-REL the implementation was based on having a distributed SPARQL endpoint that runs queries on two data sources and based on the results obtained, we can say that the Distributed SPARQL endpoint still needs more research since in most cases queries fail due to:

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 concerning the baseline (RDF materialized graph).
iii) The performance and scalability are very low when the data size increases.

For F-REL, the work was focused on including the user preferences in the queries as proof of concept to check if some value can be added to the queries in the IF architecture. The test execution principally involved the query execution with preferences and without preferences over twelve SPARQL endpoint. The proof of concept is detailed in D5.6. This proof of concept uses Skyline queries that are a kind of queries based on user preferences that identify the set of rows that are not dominated by any other row. It is considered that a row dominates another one if it is as good or better in all criteria and better in at least one criterium. The results obtained show how the average execution time of a query with preferences is very high compared to the query without preferences, this is because the query federation tool does not have optimizations designed when the data source

scales horizontally to increase the number of SPARQL endpoints. It is important to note there is still a lot of research to be conducted in the current field of construction of virtual knowledge graphs and there is much work to be done to optimize queries on functions, on the distributed approach, and latency when executing a query on multiple sources.

Finally, we expect this study 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 the transport domain. 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, and the application of query evaluation.

Performance

The main objective of this study of the performance of Distributed SPARQL Endpoint is to provide information to Improve Interoperability Framework (IF) performance to sustain a large deployment. The performance considers resources (queries and metadata) to test the capabilities of the Distributed SPARQL Endpoint and our experimental study has shown that there are still relevant open issues, such as SPARQLconformance and the application of query evaluation optimization techniques (See D3.3 sections 4.4 and D5.6). In all cases, the performance of the Distributed SPARQL Endpoint exposed the need for improvements in their current releases, in terms of efficiency and correctness of the results.

In summary, we have studied the behaviour of the Distributed SPARQL endpoint with two approaches. The first respect to query with preferences and the second, with other engines of the state-of-the-art(See D5.3) and based on the results obtained we can conclude that the Distributed SPARQL endpoint is not yet sufficiently mature and there are still relevant aspects to be addressed:

i) An important aspect is that the Distributed SPARQL endpoint does not support SPARQL 1.1 completely. For example, SPARQL queries with "FILTER NOT EXISTS" can not be resolved.
ii) In some cases, the query translation is performed naively without optimizations by Distributed SPARQL endpoint and therefore there is a need to include optimizations as part of the development of these tools.
iii) Due to the lack of maturity in the virtualization approach, no tool covers all these optimization needs in query translation and full support of SPARQL 1.1 and heterogeneous data.

Scalability

The number of available SPARQL endpoints that support distributed query processing is quickly growing; however, because of the lack of adaptivity, query executions may frequently be unsuccessful. First, the traditional optimize-then-execute paradigm may timeout as a consequence of endpoint availability. Second, endpoint query engines are not able to incrementally produce results and may become blocked if data sources stop sending data. The scalability test showcased: (i)The Distributed SPARQL Endpoint enables flexible knowledge discovery since resorts to source descriptions named RDF Molecule Templates, i.e., abstract descriptions of the properties of the entities in a unified schema and their implementation data integration. (ii) Query execution over the Distributed SPARQL Endpoint is expensive, being demanded novel techniques to generate plans able to exploit the main characteristics.

5.1.5 Converter

In the context of C-REL, we created a Converter framework (Chimera), which was then refined and improved in F-Rel, to provide a flexible solution to let developers implement conversion processes. We took into account two main conversion cases: datasets conversion and service mediation. Such cases imply very different requirements related to performance and scalability. Conversion of datasets can potentially take hours or days, and the main challenge is related to keeping memory consumption low. In service mediation the size of the messages is small, and the challenge is being able to process a message as fast as possible to be able to convert more messages in parallel. While being able to offer such performances, we also aimed at offering a flexible solution. Flexibility, in this case, means offering to developers the possibility to use different lifting and lowering techniques, and to be able to use the same framework to implement batch conversions, REST services, SOAP services, integrate with message queues, and in general being open to integration in existing production systems.

F-REL version of Chimera currently features:

- Annotation-based lifting and lowering based on ST4RT technology, which was ported to the new framework
- Declarative lifting based on RML
- Declarative lowering based on a custom solution embedding SPARQL queries inside Apache Velocity templates
- Parallel processing in both RML and ST4RT components
- Possibility to use an external RDF repository to store and query RDF triples
- Possibility to use the Asset Manager as a source of mappings, datasets and ontologies to implement a generic converter

The choice of Apache Camel as the basis for the creation of Chimera allows the integration with hundreds of components[3], further increasing the possibility to integrate a semantic-based conversion solution in a production system. After implementing batch Converters and service mediators, NAP metadata ingesters and demonstrating the possible usage of the Asset Manager to provide on-demand conversion rules, we can surely state that one of the main advantages of the SPRINT Converter is its higher configurability thanks to the modularity of the solution and the development of different blocks providing several configuration options. However, as commented in the analysis of the results in D5.6, different configurations can perform better under certain circumstances and there exist some limits that should be considered. In the following, we summarize performance and scalability considerations on the SPRINT Converter, considering testing activities performed.

Performance

The recommended solution for a batch data Converter scenario is a Chimera pipeline adopting a *declarative approach* and, in the simplest case, an RMLProcessor for lifting and a TemplateProcessor for lowering. In contrast to the *annotation approach,* which mainly targets a message mediator use case, the *declarative approach,* defining explicit rules to lift/lower data sources in a specific format to/from RDF, can be better optimised to deal also with large datasets.

[3] https://camel.apache.org/components/latest/

Considering the runtime data/message scenario, SPRINT improved considerably performance and scalability of the ST4RT converter, defining blocks for the Chimera framework that exploits the *annotation approach* and guarantee full lifting and lowering capabilities from/to a set of annotated classes. Moreover, the tests performed showcased how the *declarative approach* can effectively be used in the runtime data scenario, providing a valid alternative also for this scenario. The modularity of the Chimera solution, also enables conversion pipelines adopting the *annotation approach* only for lifting/lowering, and combining *declarative* blocks.

Considering the *annotation approach*, the results obtained demonstrated how the SPRINT Converter considerably lowered the high conversion times obtained in the ST4RT project for the same messages. Even if a detailed analysis should consider the same conversion pipeline, it is important to compare the results obtained for similar pipelines in the *annotation* and *declarative* approaches. The ST4RT approach is preferable in cases where a given data format, representable as Java classes, can be mapped without complex processing to/from an RDF representation. However, the generality of the approach, implemented without making assumptions on the source/target data format and addressing generic Java classes, results in difficulties in optimizing performances. Performance considerations on the *ConstructQueryEnricher* block, developed in F-Rel to run CONSTRUCT SPARQL queries on the RDF graph, support this claim. This block mimics the methodology followed for the START converter allowing the user to execute CONSTRUCT queries to cope with special cases not directly expressable as annotations. From a performance point of view, results pointed out that by removing the *ConstructQueryEnricher* blocks and defining a more complex logic in the lowering template, it is possible to obtain a considerable speed-up in the conversion time.

Considering the RML-based lifting, the specific RML mappings defined for a conversion pipeline (join conditions, number of triple maps, number of logical sources, path to access the records,…) can influence the performances of the lifting portion (cf. also D5.3). As commented in the previous sections, the choice of the pipeline configuration should take into account the trade-off between the conversion time and the usage of resources, for example, considering the concurrency strategies made available by the RMLProcessor. In particular, we pointed out that in some cases the gain in conversion time obtained does not justify the higher resource usage (small batch datasets and JSON data format). Finally, we specify here additional recommendations that emerged in our experience in testing the RMLProcessor:

- to efficiently exploit concurrency it is also important to tune the different parameters, e.g., the number of concurrent threads allowed;
- concurrency should not be used in case of mappings defining blank nodes that are referenced by different triple maps (each thread would assign different random identifiers to the same blank node);
- the presence of many *functions* in the RML mappings can cause concurrent access to the same data structures limiting the conversion time speed up;
- concurrency strategies can be implemented not only in RMLProcessor but also in the pipeline, for example, configuring different RML blocks in parallel or exploiting concurrent consumers for Camel routes.

Considering the lowering portion based on templates, we discussed in details in D5.3 how the queries and the logic adopted in accessing their results can heavily affect the performances. As described in D5.5, for F-Rel we implemented a *stream* option to process templates in-memory (avoid

input/output operations) that improves performances for the runtime data/message conversion use case. It is recommended to avoid using this option for large batch datasets because the Template Engine, without this option, optimizes the memory consumption writing incrementally the output to the filesystem.

Scalability

Considering the scalability of the proposed solution, it is important to discuss in particular the batch data conversion scenario. In C-Rel, we pointed out the memory consumption problem related to the materialization of large knowledge graphs while assessing that the virtualization techniques are still not mature enough to be employed in production systems.

In the developments for F-Rel, we tried to cope with this problem by improving the RMLProcessor implementation and implementing the possibility of using an external repository to store the materialized graph. In the performed tests, we showcased how these approaches can reduce memory consumption but still have some limits. In particular, the use of an external repository shifts the bottleneck to the RDF repository, which in many cases cannot keep the pace in indexing a large number of triples, even considering incremental writes. For this reason, for the conversion of very large datasets, it is recommendable to split the procedure under the assumption that the materialized graph does not change very often:

1. execution of the lifting procedure (if required splitting the mappings in different executions);
2. adoption of a tool for bulk loading of the materialized graph in an external repository[4] to avoid indexing issues;
3. define a Chimera pipeline for an on-demand lowering of the materialized graph.

This type of approach also allows users to select different tools for lifting. Indeed, the RML specification has several implementations[5] that, considering the requirements of the specific scenarios, can offer better performances as shown for the SDM-RDFizer[6] in D5.3. The RMLMapper[7] adapted in SPRINT to implement an optimized RMLProcessor, is constantly updated to improve performances and reflect possible modifications in the RML specification, future implementations of the Converter should be aligned to the latest release.

Finally, as described in D5.5 in some cases it is not needed to materialize the entire knowledge graph from the input data sources to obtain the conversion. In these cases, the pipeline can be optimized considering a subset of the RML mappings for the lifting portion identified taking into account the lowering queries defined in the template.

[4] For example, for GraphDB https://graphdb.ontotext.com/documentation/standard/loading-data-using-the-loadrdf-tool.html

[5] The RML implementation reports lists which test cases for the RML specification are covered by each implementation https://github.com/RMLio/rml-implementation-report

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

[7] https://github.com/cefriel/rmlmapper-cefriel

Considering the scalability for the runtime data/message conversion scenario, the testing activities showcased: (i) the improvements obtained for the *annotation approach* (previously not allowing concurrent processing of requests), and (ii) how the *declarative approach* can offer very good scalability results even considering a single instance of the Converter.

.

5.1.6 Collaborative ontology manager

For F-Rel we created a Collaborative Ontology Manager. The collaborative construction of ontologies has become a central paradigm of modern ontology engineering. This understanding of ontologies and ontology engineering processes is the result of intense theoretical and empirical research within the Semantic Web community. That is why in the context of Shift2Rail, collaborative development, it is generally recognized that, in order to be useful, but also economically viable, ontologies should be governed, developed and maintained in a community-led manner, with the help of comprehensive environments that provide dedicated support for collaboration and user participation.

Collaborative ontology manager, as described in detail in D4.3, is a proof-of-concept tool able to work with several types of version control systems (tested on platforms like GitHub, GitLab and Bitbucket), obtaining good results in the documentation generation and quality evaluation of the ontologies. This tool applies mechanisms such as pipelines with continuous integration tools (e.g. Jenkins) where each user can create a task, add a configuration file (Jenkins file) inside the repository where the ontology is located and automatically deploy all the workflow.

Although it does not main affect the performance of the IF architecture, we have realized a performance and scalability study to be able to show the effect that it could have when integrating these tools to automate and accelerate the development process of the ontology. We observed that the **performance** process of both the evaluation and the generation of documentation obtained of the principal steps after the execution of the tool over an ontology is approximately a few seconds. In the **scalability** test, we evaluate quality for multiples ontologies to measure how the input of multiple ontologies and their growth in the number of concepts affects the performance of the documentation generation and evaluation process, this allows us to make an approximate measure of the time of generation of documentation and evaluation as the file grows at the level of classes and relationships.

5.1.7 Automation in ontology development

In the context of Shift2Rail, we want to automatically generate conceptual models from semi-structured models. The automation of ontology development from existing XML Schemas can speed up and simplify the match and merge processes with S2R ontologies. XSD2OWL tool allows the automatic transformation from the XML Schema to OWL by means of the integration of many XML data. For F-REL, we will focus on transforming representations of the XML schema components of NeTEx.

XSD2OWL[8] can be applied for XML semantics reuse and it is based on mapping from XML Schema constructs to the OWL ones that are semantically more appropriate. XML schemas are used in grammars as the source from which the semantics they capture implicitly are going to be formalized

[8] https://github.com/oeg-upm/sprint/tree/main/xsd2owl

and made explicit. In general, the transfer of XML metadata to the ontology is not made explicit when XML metadata instantiating these schemas is mapped.

To simplifying and automate the necessary steps needed to integrate new services and sub-systems in the IP4 ecosystem we analyzed the **performance** such as execution time and memory consumption for transforming each XML Schema file from NeTEx to OWL. **Scalability** shows the test over the number of simple and complex types that have each XML Schema in the transformation to an OWL file. The memory consumption of our test has not exceeded 1MB (see D5.6).

6. RECOMMENDATIONS FOR THE IP4 IF SEMANTIC AUTOMATION

The Interoperability Framework supports by design a wide array of automation solutions, and it is being used as a starting point to implement an IP4 ecosystem. To that extent, the IP4 ecosystem (as shown in Figure 2) is just one of the possible interoperability solutions which can be implemented using the IF.

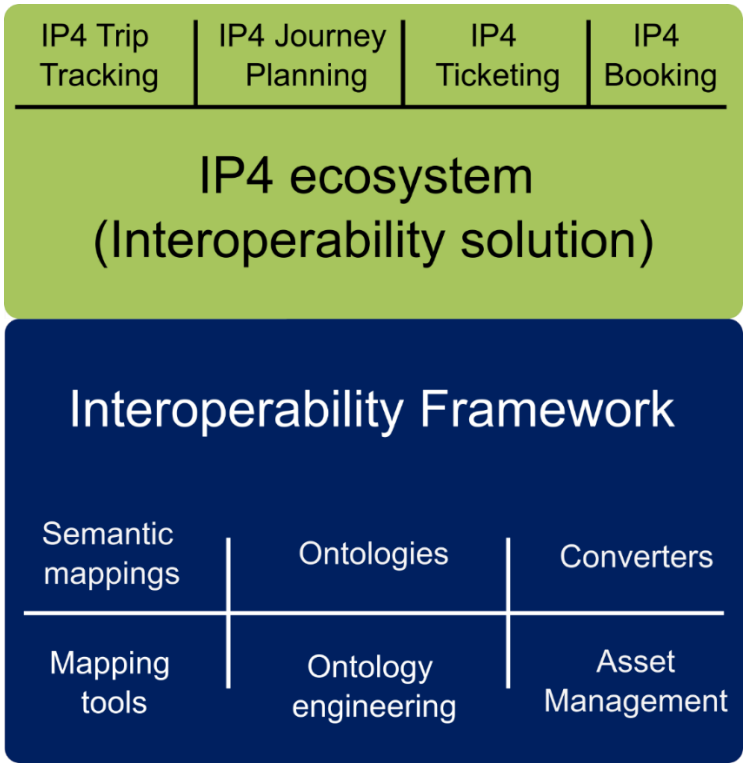

Figure 2: Interoperability Framework and IP4 ecosystem

The establishment of an IP4 ecosystem encompasses several different activities which must be taken care of. Such activities stem from the creation of a common model to the development of the integration solutions to the usage of sharing platforms to raise awareness and ease the adoption. A high-level list of activities is reported here:

1. Develop the ontology;
2. Analyse existing ontologies;
3. Develop mappings;
4. Develop converters;
5. Develop resolvers;
6. Publish artifacts;
7. Gain access to existing artifacts;
8. Perform tasks after the successful publication of artifacts.

Automating parts of this long list of activities can play a key role in establishing an efficient ecosystem, lowering the maintenance costs and helping govern a complex distributed system ran by different transport operators. Some of those activities can be fully automated, while others require human intervention. In the following sections, we will describe the possible roles of automation and provide a set of recommendations to the rest of the Shift2Rail IP4.

6.1 ONTOLOGY DEVELOPMENT

Ontologies are being increasingly adopted in information systems, it is clear that ontology development tasks may also benefit from the application of common software engineering practices. Most of the ontology development support activities, such as documentation, visualization and evaluation, are usually performed individually, executing heterogeneous tools that make these activities cumbersome and time-consuming. Besides, maintaining and keeping track of the generated resources for each version of an ontology has become a challenge for ontology developers.

Ontology development is a complex subject, and we can roughly divide it into two main branches: creating a new ontology from scratch and converting an already existing data model using the W3C Semantic Web stack. The former activity requires a fully human activity since it involves understanding a domain, its rules and representing them as a set of logical axioms. In this sense, there are has already shown progress towards adapting ontology development to agile software development methodologies, continuous integration of support activities when new changes in the ontology are registered as well as supporting collaborative ontology development throughout the use of common-practice software engineering tools. For example, it is now common among ontology developers to use Git-based environments such as GitHub[9](usual in software development) for keeping track of ontology revisions. Automation is any way possible to support human activity, easing collaborative editing and providing up-to-date graphical representations of the ontology. Those two aids are especially useful when the domain which is being modelled is vast, and when the team is actively working at different aspects at the same time.

Full automation is not possible even in the latter branch of ontology development, namely the conversion of an existing model into an ontology. In this case, the conceptualization of the domain has already been performed by someone else, but it has been serialized into a non-ontological format. XML or RDB schemas, UML diagrams, are all examples of such non-ontological formats. The aim of the ontology development activity, in this case, is to obtain a clean model, removing attributes and relations that are usually introduced by the specific format, while at the same time staying very close to the original model to keep compatibility. The role of automation, in this case, is to provide a first rough draft of the ontology, which can be used by ontology designers to speed up the development process since such ontology designers need a first rough draft and human-readable documentation to understand the taxonomy and relations included in a vocabulary and assess whether it addresses their requirements.

While converting messages and connect different systems, the quality of mappings is a critical issue. Such quality is influenced by many factors, like the level of knowledge of the domain, knowledge of the two ends of the communication channel, and also changes in the models during the time. The

[9] https://github.com/oeg-upm/transmodel-ontology

semantic-based solution can help in identifying missing data while developing mappings, and in identifying incompatibilities generated by changes in the ontologies. SHACL is an RDF-based language useful for data validation while converting messages. This technology can be used both to detect missing data during conversion and to drive the development of new mappings. Its role in the Semantic Web stack is akin to XML Schemas in the XML stack since it allows specifying cardinalities and consistency rules for RDF data. The development of SHACL shapes could be included in the ontology engineering efforts, and releasing SHACL shapes with each ontology release could help mapping developers in providing better Converters. Ontology changes during time is another factor affecting mappings quality. Each new version of an ontology could break existing Converters, and early detection of incompatibilities will be a key factor in keeping the soundness of the IP ecosystem. The development of test cases could help to tackle such issue, and such test cases could be automatically executed by the Asset Manager after publishing a new version of the IP4 ontology. Test cases for the IP4 ecosystem should imply the following requirements:

- Each Ontology should define a set of queries (or competency questions) that can be answered
- Each Ontology should publish an example dataset
- Each Converter should declare a sample input and output message, and the ontologies used during the conversion

This set of requirements would allow a test to notify Converters and Mappings owners that changes in the ontology structure are going to break their artifacts, in case they are developed to use the latest version of the reference Ontology.

Summary of recommendations:

- Provide up-to-date diagrams of the Shift2Rail IP4 ontology;
- Provide documentation about possible alignments with existing ontologies;
- Release SHACL shapes together with the Shift2Rail IP4 ontology to show how the ontology is intended to be used;
- Use competency questions to create tests;
- Provide examples of how the Ontology can be used (data samples);
- Provide input and output examples for Converters;
- Link a Converter to Mappings and Ontologies in the Asset Manager;
- Integrate into the life cycle of coarse-grained support activities involved in ontology development, such as documentation[10], versioning, evaluation[11] and publication of ontologies that are maintained and versioned in a Git-based environment.
- Use an open-source continuous integration automation server such as Jenkins to automatically generate the documentation and evaluation in the development life cycle of the ontology establishing an efficient ecosystem.

[10] https://github.com/dgarijo/Widoco

[11] http://oops.linkeddata.es/

6.2 INTEGRATION ENACTMENT

The current efforts of the SPRINT and CONNECTIVE projects are aiming at providing and testing interoperability solutions based on the concept of the Interoperability Framework. Integrating different systems requires overcoming multiple technical difficulties, and the integration process requires analyzing several aspects, like:

- whether the two systems use the same communication approach (pull vs. push);
- whether the two systems are stateful or stateless;
- whether the two systems use compatible processes;
- how much information from a source system can be sent to the destination system.

The SPRINT project is demonstrating that some aspects of the integration process can be streamlined using a combination of semantic techniques and already existing open source solutions. When integration is a message-to-message conversion, the Asset Manager can automatically generate a running Converter by just stating the relevant ontologies, dataset and mappings. Even if we showed that automation can be fully applied, the case of a message-to-message conversion is any way just a small subset of the cases in the domain of the transportation domain. Past work in the ST4RT project demonstrated that the biggest obstacle in integration is the different information granularity between two systems, and the case of FSM to TAP/TSI 918 showed that message exchanges are usually part of larger processes. In such cases, it is important to store the context information which is attached to the process instance.

In all the cases where a simple message-to-message conversion is not feasible, a possible and recommended solution is to take the simple case as a starting point and to create customized conversion pipelines leveraging on the components supported by Apache Camel, which is at the core of the SPRINT Converter solution. In cases where the processes to be implemented are complex, a viable solution that minimizes manually coding is to embed a Business Process engine inside the Converter. With that solution, a large part of the process mediation could be implemented by drawing BPMN diagrams, while the actual conversion of messages would be delegated to the semantic components already supported by the SPRINT Converter framework (Chimera).

Use case Scenario 10, described in D5.4 and demonstrated in D5.6, contains a valid recommendation related to building a scalable interoperability solution based on the outcomes of SPRINT. The scenario shows how the Asset Manager can be used as a configuration management server for the Converter. We implemented a Chimera configuration that is able to dynamically fetch new conversion routes from the Asset Manager. With such configuration, whenever the Converter receives a conversion request between two unknown formats, specifications or standards, it dynamically checks whether a valid instance of a “Converter” asset type exists containing both ends of the conversion pipelines. If the asset is found, then according to its metadata the Converter fetches from the Asset Manager all the ontologies, datasets and mappings required to properly execute the conversion, and processes the request. From that moment on, the Converter will know such conversion route and will not ask again the same information to reduce traffic, since the assets inside the Asset Manager are not updated so frequently. This interaction enables the possibility of considering a scalable interaction enactment, where a variable number of “generic” Converters are deployed together with a single instance of the Asset Manager. In such deployment adapting the number of replicas of the Converter in the deployment environment would be easier to implement, as there would not be any specific metric to be gathered other than the meantime to process the

requests or the mean CPU utilization. The only limitation of such an approach is related to the specific data transformations to be implemented. Our pre-made conversion pipeline takes into account a single RML lifting and a single Velocity+SPARQL lowering template. This means that some cases may exist where such a pipeline is not powerful enough to fulfil the conversion requirements. Stateful conversions or complex conversion processes involving the orchestration of multiple external API are just two cases which could not be handled using our “generic” approach. Those cases would require a specific conversion pipeline which would have to be implemented by means of a custom Converter configuration using our Chimera framework.

Summary of recommendations:

- Document the list of features of each system that is being integrated inside the IP4 ecosystem;
  - Pull-based vs. push-based
  - Stateful service vs. Stateless service
  - Processes involved with each message exchange
- Use the basic message-to-message conversion pipelines automatically generated by the Asset Manager as a starting point to create stateful or process-based Converters;
  - In case the processes to be mapped are complex, consider embedding a Business Process engine inside the Converter
- As an alternative providing a general scalability model, use the Asset Manager as a configuration management server for Converters who will be able to dynamically obtain new conversion routes.

6.3 ECOSYSTEM MAINTENANCE

As introduced before, the IP4 ecosystem is an interoperability solution that is being built for the transportation domain using the tools of the Interoperability Framework. Some of the tools provided by the SPRINT project can be used to efficiently address ecosystem maintenance. With the term “ecosystem maintenance” we mean all the activities which contribute to keeping a distributed system running with minimal inconsistencies and downtimes. Such activities encompass the high-level governance of the ecosystem, dealing with rules and contracts between companies, and the constant checking that the information contained in the asset descriptions is up to date and does not lead to an inconsistent distributed system. The Asset Manager can play a key role in automating all the technical parts of the governance process since its role is of being the single source of knowledge for all the components of the ecosystem. It can, therefore, ensure that a major release of service is notified in time to all its users, it can be used to automatically execute ontology-based tests to ensure that changes do not break any existing Converters, and it can be used to automatically convert datasets into other formats (like the Transmodel-based standards required by National Access Points).

Exploiting the automation functionalities of the IF as implemented by the SPRINT project requires defining clear governance of the IP4 ecosystem. Such governance should define roles and responsibilities, and should also define lifecycle management processes for the various types of assets that are to be managed by the ecosystem. Once such processes are defined, they can be

drawn and developed using BPMN and deployed onto the Asset Manager, which will then be able to enact such governance structure and automate its effects.

As shown in Figure 2, the tools developed in SPRINT are meant to support the creation of an IP4 interoperability ecosystem. Such an ecosystem will allow the developing of higher-level features such as shopping orchestration, trip tracking or journey planning. If such IP4 ecosystem is meant to be an “open” ecosystem where different service providers can develop and offer their solutions, the APIs of the high-level features should be published on the Asset Manager as well as the underlying data model (the IP4 Ontology).

Summary of recommendations:

- Define a governance structure for the ontology;
- In case of a centralized deployment of the IF, define a governance structure and IF ownership;
- Define and share the details of lifecycle management processes;
- Pay attention to the dependencies between assets to avoid breaking the IF functionalities;
- Use the Asset Manager as a command and control centre, linking the lifecycle management to automation tasks to be performed after a successful publication;
- Document all the APIs of the high-level components of the IF (Shopping orchestrator, Journey planning, …) using a machine-readable format;
- Publish the APIs of the high-level components of the IF in the Asset Manager;
- Provide mappings from and to the IP4 Ontology and the data structures used in high-level components of the IF.

7. RECOMMENDATIONS FOR THE MARKET UPTAKE

Chapters 4 in both SPRINT Deliverables D5.3 and D5.6 “Validation of the pilot implementation” C-REL and F-REL, respectively, describe the fundamental criteria adopted in the conceptual design and prototype design of the IF to address Indicators for its evaluation based on the recommendations and for the IF development and deployment produced in GOF4R D5.2 “Toolkit of recommendations and KPI Scoreboard”. The rightmost column in the following table summarized them:

Indicator	Description	IF contribution to addressing the 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.	The IF should facilitate the 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	The critical mass can be gained through minimizing or eliminating the need for adaptation of legacy systems and participation in centralized governance. The mandatory requirement to join the IF ecosystem has to be limited by the necessity to register. The way how the IF should work: availability to publish resources and discovery of other stakeholder resources through the Asset Manager.
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.	Create the same rules and requirements for all stakeholders by minimizing ICT development costs and governance overhead through minimizing or eliminating the need for adaptation of legacy systems.
Stakeholder’s management	One of the key elements that the IF governance has to address: lack of cooperation among stakeholders, collaboration with other non-transport related entities (organizing authorities,	“lack of collaboration” should be considered from a point of view of making interoperability a digitalized mechanism instead of a ‘governance’ policy. The focus should be on registering/publishing the resources/assets in the Asset Manager keeping full control of them, and

Indicator	Description	IF contribution to addressing the indicator
	financial institutions, IT services,…)	discovering available resources registered/published by other stakeholders. In this scenario, the nature of governance is therefore changed to maintaining the collaboration *tools*, i.e. the IF components themselves.
User-friendliness	To what extent the solution addresses the issue of lack of knowledge in semantic ontology and data-interoperability	The solution has to use standard semantic web specifications and other standard technologies and should be architected to allow the flexible composition of processing chains from common blocks through configuration scripts of open-source build and runtime frameworks., minimizing the need for ICT development.
Reliability and security of the ecosystem	How the solution addresses the issues on cybersecurity.	Reliability and security should be 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.

As described in chapter 4 of the SPRINT Deliverable D.5.6, the suitability of this approach has been evaluated in a significant context, namely the incorporation of National Access Points (NAPs) in the ecosystem and conjunction with the CONNECTIVE project. This scenario is important in that NAPs are existing external systems that support regulatory provisions and are implemented in different architectures, technologies and capabilities in the different European Member States, constituting an environment that is 'given' and cannot be 'adapted', but must, by regulation, be part of the ecosystem: it is an environment fairly representative of the challenges faced in the dynamic construction of the ecosystem in the presence of external technical, organization and regulatory constraints.

The main finding of the evaluation is that there is no *single* interoperability problem amenable to a *single* interoperability solution, but that a *common* collection of specialized tools providing specific capabilities must be made available to ecosystem stakeholders in order to compose different interoperability solutions for the particular interoperability problems arising from specific environments. The common collection of tools must be developed according to an architecture that leverages standard languages and frameworks, and that separates application (business) level logic from the mechanics of 'pure' interoperability, delegates security and reliability provisions to the underlying runtime environment, and permits deployment in multiple instances of multiple runtime environments.

From this finding, and in addition to those listed in the preceding chapters, the following recommendations for market uptake can be derived:

- Release the different tools as an IF ‘suite’ under an open-source license, engaging the ‘open community of technology specialists in their further improvement and extension, porting to different frameworks and underlying environments;
- A dedicated ‘development’ instance of the Asset Manager may be used in conjunction with platforms such as GitHub to manage the development/testing process under IF tools development governance processes as described in the outcomes of the GOF4R project;
- Establish a training and support team as part of the IF Governance to assist end-users in the composition and utilization of the tools in the ‘suite’ to design specific interoperability solutions for their interoperability problems, providing automated debugging/testing tools, documentation, on-line help and validation of the composing solution.

8. CONCLUSIONS

This deliverable D2.5 is the final version of recommendations to Shift2Rail IP4 issued after the validation of the F-REL proofs-of-concept.

Proposed recommendations cover different aspects:

1. Recommendations based on high-level requirements' analysis of S2R IP4 projects and related EU initiatives in the framework of SPRINT D2.1. The recommendations are proposed to three main points:
    - Leveraging automation
    - Monitoring and Governing
    - Leveraging the technology neutrality
2. Recommendations for the IF architectural design.
3. Recommendations for the IF architecture as a component to NAP and includes recommendations related to:
    - Metadata interoperability
    - Data compliance
    - Accessibility and contributing datasets
4. Recommendations for performance and scalability of the IF, including the lessons learned from F-REL implementation and validations for the asset manager, user manager, mapping tool, converter, collaborative ontology manager, and automation in ontology development.
5. Recommendations for the IP4 IF semantic automation, including ontology development, integration enactment, and ecosystem maintenance.
6. Recommendations for the market uptake based on the indicators defined in GOF4R D5.2 'Toolkit of recommendations and KPI Scoreboard'

These recommendations can be applied in further work on the IT interoperability framework for public transport.

9. REFERENCES

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

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

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

[4] European Commission, Directorate-General for Informatics, "New European interoperability framework : promoting seamless services and data flows for European public administrations," Publications Office of the European Union, Luxembourg, 2017.

[5] SPRINT project, “D2.1 Initial analysis of requirements of S2R IP4 projects and other EU initiatives,” 2019.

[6] SPRINT project, “D2.2 Requirements for an IF architectural design (C-REL),” 2020.

[7] SPRINT project, "D2.3 Requirements for an IF architectural design (F-REL)," 2020.

[8] SPRINT project, "D3.2 Performance and scalability requirements for the IF (C-REL)," 2020.

[9] SPRINT project, "D4.2 A lightweight solution to automate the generation of ontologies, mappings and annotations (C-REL)," 2020.

[10] SPRINT project, "D5.1 Requirements, scenarios and use cases for the proof-of-concept (C-REL)," 2020.

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

[12] 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.

[13] GOF4R project, "D5.1 - Deployment Roadmap," 2019.

[14] GOF4R project, "D5.2 - Toolkit of recommendations and KPI Scoreboard," 2019.