FAIRVASC
Onboarding Handbook

Version 2

FAIRVASC Onboarding Handbook

Version History

Version Number	Date	Comment
1	16 Jan 2023	Initial version provided to onboarding registries ahead of FAIRVASC Onboarding registries meeting, Krakow, Poland, 20 Jan 2023
2	28 Aug 2023	Started 17 Apr 2023 as a working document

Table of Contents

1 Summary

This document introduces the background and aims of the FAIRVASC project, a description of the concept of the Semantic Web and FAIR Principles, and practical "how to" guidance for onboarding of a registry to the FAIRVASC consortium. It is aimed at both registry clinicians and computer scientists and requires no previous knowledge of Semantic Web technologies or rare disease research.

2 An introduction to FAIRVASC

FAIRVASC is a research project bringing together computer scientists, clinicians, and patient organisations to expand our knowledge of the group of diseases 'anti-neutrophil cytoplasmic antibody (ANCA) associated vasculitis' (AAV). As AAV is a group of rare diseases, the patient numbers needed for rigorous clinical research are too low in any given country.

To tackle this challenge an exchange of data between researchers, organisations, and countries is needed. Patient registries are a valuable research resource as they collect information on the disease in terms of patient's demographic and clinical history variables, yet today AAV registries remain scattered and unconnected. Of the data that is digitised, it is rarely standardised from the perspective of data interoperability, and data analytics over multiple registries is not possible as the structure and semantics of the data differ. Thus, the shared information contained in the registries is not used to its full potential.

In FAIRVASC we aim to open the door to new research in AAV by using semantic-web technologies to link the registries included in the project into a single large dataset, creating a dataset of unprecedented size.

2.1 The Disease

Anti-neutrophil cytoplasmic antibody (ANCA) associated vasculitis, or AAV, is a group of rare diseases sharing clinical and pathological features. In patients with AAV the immune responses are misdirected against the body's own healthy cells and tissues, resulting in inflammation and destruction of small blood vessels.

The name stems from the group of diseases associated with a type of autoantibodies (antibodies directed against an individual's own proteins) called anti-neutrophil cytoplasmic antibodies or ANCA. There are two main ANCA serotypes, based on the particular protein ANCAs are directed against, namely anti-PR3 (autoantibodies against proteinase 3) and anti-MPO (autoantibodies against myeloperoxidase). Although some researchers prefer to classify AAV into anti-PR3 and anti-MPO associated disease, the most used sub-classification is based on the clinical presentation as: granulomatosis with polyangiitis (GPA), microscopic polyangiitis (MPA) and eosinophilic granulomatosis with polyangiitis (EGPA).

The symptoms vary based on the disease type and may range in severity from non-specific flu-like symptoms to life-threatening organ failure. GPA commonly involves the upper and lower airways and shows a stronger association with anti-PR3, while MPA commonly affects the kidneys and has a stronger association with anti-MPO. EGPA is a type of vasculitis that usually occurs in patients with asthma and allergic rhinitis. Despite being called

anti-neutrophil cytoplasmic antibody-associated vasculitis, ANCAs are not always present, and ANCA-negative AAV may occur.

As the immune responses are misdirected in AAV, the treatment is based on suppression of the immune system. The drugs used include corticosteroids, cytotoxic drugs, and biologics.

The prognosis and quality of life for patients with AAV is often poor. It has however improved remarkably after the introduction of effective treatment. However, the often needed prolonged use of immunosuppressive drugs to keep symptoms at bay may lead to serious toxic side effects such as malignancies, infections, and metabolic complications (for example diabetes).

There remain many unknowns in AAV. Much of the research is hampered by the low patient counts, as rigorous clinical research relies on large sample sizes. By federating registries, FAIRVASC is a stepping stone to shedding light on the unknowns of AAV.

2.2 FAIR Principles

The FAIRVASC project is built on the foundation of the FAIR Principles, first described in 2016. The FAIR Principles are guidelines to improve the Findability, Accessibility, Interoperability, and Reuse of digital assets, such as a disease registry. In 2018 the GO-FAIR initiative provided a framework for the local implementation of the FAIR Principles. This ‘FAIRification process’ applies to the metadata (data about the data), the data and the supporting infrastructure of, for example, disease registries.

The FAIRification process utilises Semantic Web technologies to link data and make it accessible via the web. You can read more about the FAIRification process and the FAIR Principles at: https://www.go-fair.org/

2.3 The Semantic Web

The world-wide-web was initially conceived as a web of documents rather than a web of data, as it was intended for human use. When reading a document, it is the previous knowledge of the context that allows us, as humans, to interpret and understand the meaning of its content. This, however, is a human prerogative and it's not possible for machines. The concept of the Semantic Web (where the term "Semantic" comes from the Greek word for "meaning") was developed to make the information in the web of documents more meaningful to both humans and machines, thus creating a web of data. Instead of relating documents, the goal is to relate data contained in the documents with explicit meaning or semantics, to create a Web of Data, rather than a web of documents. As such, the Semantic Web provides a framework that allows data to be shared and reused across application and community boundaries.

The key to this is to use Internationalised Resource Identifiers (IRIs) to name or identify anything that needs to be described on the web. Much in the way that a Uniform Resource Locator (URL) is a ‘web address’ which provides a unique web identifier for a website, an IRI can be considered a unique ‘data address’. An IRI contains information on the resource, the location of that resource and contains the identifier of what you want to describe. An example is the IRI: <http://w3id.org/FAIRVASC#ANCA>, where the resource and location of the resource is <http://w3id.org/FAIRVASC>, and the identifier is <#ANCA>.

IRIs are the foundation of the Semantic Web and the encoding of semantics with the data. To encode semantics, technologies such as Resource Description Framework (RDF) are used. RDF is a data model designed to be read by computers and uses IRIs to identify and reference resources on the Web.

RDF is a graph data model as a general method for description and exchange of graph data. Importantly in this context, a ‘graph’ does not refer to the type of diagram clinicians would be used to seeing in a medical journal, such as a bar chart or a scatter plot. In the computer science world, a graph is a way to represent information as a network of interconnected nodes. Connections between nodes are called edges. As the purpose of the RDF data model is to represent the data along with its meaning, a subject-predicate-object statement (called a triple due to having three components) is used as the base unit to represent any data. Each of the three parts of the triple statement can be identified by an IRI. One can also call the subject and object ‘nodes’ and the predicate an ‘edge’.

A traditional relational database presents data in tabular form, where each table consists of a set of rows and columns. In, for example, an AAV registry it would be common that each row represents a patient with a unique id and each column represents a specific attribute, for example the diagnosis. In the intersection of the row and column (the cell) the value of the specific attribute for the patient would be presented, or the type of diagnosis.

id	diagnosis
001	Granulomatosis with polyangiitis

A graph representation of the same information is based on triples. Patient of id 001 (subject) has diagnosis (predicate) of granulomatosis with polyangiitis (object). Preferably, the object is encoded according to a standard terminology to further enable interoperability. However, the object as opposed to the subject and predicate is not required to be an IRI. An example containing the same information as the relational table above in RDF, utilising a standard terminology, could be written as:

<http://example.org/patient/001>

<http://example.org/hasDiagnosis>

<http://identifiers.org/orphanet:900>

A database of graph data (RDF) is composed of multiple triples. As opposed to a relational database, a specific data property may be a subject, predicate or object depending on the context, creating a network that may be visualised as a graph. A database containing graph data is called a triplestore. Multiple triplestores may be interconnected and made searchable remotely in federated fashion via the web through exposed endpoints. SPARQL is a language to write queries and retrieve data from a triplestore.

More information on the concept of the Semantic Web and hands-on training in creating RDF and writing SPARQL queries, is available in the FAIRVASC training material (see section 5 for link).

3 FAIRVASC Technical Details

3.1 The FAIRVASC Ontology

In the field of information science, an ontology can be thought of as a way of showing the properties of a particular subject area and how they are related by defining a set of concepts and categories that represent the subject. To achieve this, ontologies make use of the Semantic Web and RDF, as described in the previous section.

The FAIRVASC ontology is the core of the FAIRVASC project. All the shared concepts of the registries in the FAIRVASC collaboration and their relationships are described in the ontology, which can be viewed (somewhat simplified) as the vocabulary or dictionary of FAIRVASC. You can find and browse the latest version of the FAIRVASC ontology at http://w3id.org/FAIRVASC.

The FAIRVASC ontology is connected to other standard terminologies and ontologies, such as Orphanet Rare Diseases Ontology (ORDO) and the National Cancer Institute Thesaurus (NCIT), to describe concepts in a uniform fashion.

The creation of the FAIRVASC ontology is an ongoing iterative process where clinicians and computer scientists work together to harmonise the data in the different registries to a common semantic structure.

3.2 The Semantic Uplift

In the Semantic Uplift registry data contained in relational tables are transformed into RDF. This is achieved through the declarative mapping language R2RML. With R2RML one can declare how data from a non-RDF resource (for example a table) should be transformed into RDF. This process can also be called mapping.

To achieve a uniform FAIRVASC data structure, each registry maps their data to match the FAIRVASC ontology. This creates a new, RDF version of their registry, while the original data remains untouched.

The last step to make the data findable, accessible, interoperable, and reusable, is to make the registries interconnected and searchable as one. Each registry site deploys a local triplestore, storing their RDF data. Each triplestore has an endpoint exposed. This enables the triplestores to be reached through a website, the FAIRVASC interface. Through the FAIRVASC interface, vasculitis researchers can pose pre-defined queries over the RDF data, and retrieve results, without any subject level data leaving the registry sites.

3.4 The FAIRVASC Cycle

The development of the FAIRVASC ontology and the Semantic Uplift is centred around the interaction of three work-teams: the Query Implementation Team (QIT), the Harmonisation

Implementation Team (HIT) and the FAIRVASC Implementation Team (FIT). Each team has at least one representative from each registry in the FAIRVASC collaboration.

QIT is a team of experts in the field of vasculitis that drive the requirements for data harmonisation through the development of competency questions. The competency question informs HIT, a team of experts in the local registry structure with additional clinical knowledge. A HIT member could be, for example, a clinician with experience in adding patients to a registry. HIT identifies the variables in the registries needed to answer the competency questions and tries to find a common, shared term across the registries. When this term has been found they inform FIT, a team of computer scientists and data managers, who implement the terms in the ontology and in the Semantic Uplift. Lastly the FIT implements the competency question as a machine-readable query on the FAIRVASC interface, using SPARQL.

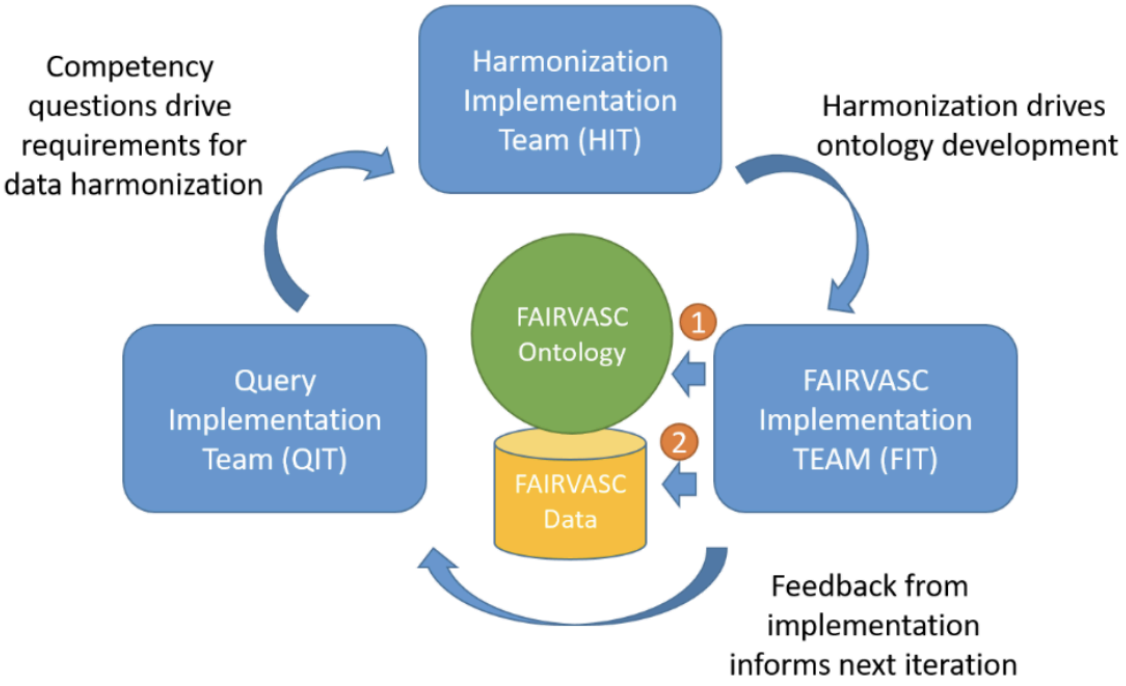

Figure 1 *Overview of the interaction between the FAIRVASC teams. The FIT implements the suggestions from the QIT and HIT in a two-step approach of 1) Creating the FAIRVASC ontology 2) Mapping the registry data to the format described in the ontology using the declarative mapping language R2RML.*

4 A Practical Example Of The FAIRVASC Cycle

Original registry data in the FAIRVASC registries differ in respect to semantics and structure. Data may be collected retrospectively or prospectively, on paper or directly via an electronic clinical report form and be exported for analysis in a variety of file formats such as .xlsx or .csv. However, an important feature in all registries is the data dictionary. In the data dictionary the variable names, field types, field labels, and choices are described.

Through the data dictionaries the information contained in the different registries can be explored without exposing subject level data. The first step of harmonisation is to become familiar with the terms available in the registries, and how they align with the competency question. Let's start with a simple example:

1. QIT develops a competency question. The experts in AAV across the consortium are interested in knowing: *How many patients with granulomatosis with polyangiitis (GPA) are there across the FAIRVASC registries?*
2. This competency question is picked up by HIT. HIT explores the registry data dictionaries for the information needed to answer the question. This is done by searching for the variable names and choices containing the information needed and how they align across the registries. Examples for how the different registries in the FAIRVASC consortium code for GPA are presented in table below:

Registry	EUVAS	RKD	GeVas	Czech	Skåne	PolVas	FVSG
Variable name	small_vessel_vas_anca	small_vessel_vas_anca	VD_VASC5	dgema	diagnosis	1.16	DIAGN
Choice	900	3	1	1	1	Ziarniakowatość z zapaleniem naczyń (Wegenera)(GPA)	6

As you see, the information is available in all registries and aligns well (it can be assumed that the information means the same across all registries).

HIT then set out to find a suitable term for the diagnosis of GPA for the shared vocabulary of FAIRVASC. To ensure that the FAIRVASC ontology is sustainable and made interoperable, it is preferable that the term used is universally accepted. This is done by using standard terminologies. In the case of diagnoses, FAIRVASC is using the Orphanet Rare Disease Ontology (ORDO). The unique ORDO identifier for Granulomatosis with Polyangiitis is the code: 900.

However, the harmonisation may be less straightforward. In some cases, ontology development means trade-offs between more generic terms, covering all registries and more specific terms that may cover only some registries, but which may provide more meaningful information to researchers. Hands-on training in harmonisation of clinical terms is available in the FIT training material.

3. This information is passed on to FIT, the technical team of FAIRVASC. FIT first develops the FAIRVASC ontology to include information about the main diagnosis of the patient. When the ontology is expanded, each local registry maps their registry information to match the FAIRVASC ontology using R2RML. The diagnosis of GPA will be presented as an IRI pointing to the ORDO identifier 900: (http://identifiers.org/orphanet:900). A new RDF subset of the registry encompassing information about the diagnosis of each patient is created. This RDF version of the registry is uploaded to the local triplestore.

Lastly, FIT implements a query about the diagnosis on the FAIRVASC interface, using SPARQL. This query, presented to the end-user as a check-box option on a website, retrieves information about the number of patients with the diagnosis of *Granulomatosis with polyangiitis* across all the FAIRVASC registries. The team of vasculitis experts in QIT, can

now retrieve this information. The process starts over, when QIT meets up to discuss a new competency question of interest.

Hands-on training-material in ontology development, R2RML, and SPARQL are available in the FIT training material.

5 FAIRVASC Training Material

The training material provides hands-on practice in ontology development, the harmonisation of clinical terms, R2RML mappings, and the writing of queries in SPARQL. The FIT training sessions are aimed primarily at the FIT personnel at your registry site. However, it requires no previous experience in programming and may also be suitable for the interested clinician or HIT representative.

There are three FIT training sessions available, estimated to take 10 hours in total. The sessions (available as pre-recorded videos) and the session materials are available at:

Training Material

6 Data Governance

It is essential that FAIRVASC complies with data governance legislation, such as GDPR and member state law. To achieve this a project-level Data Protection Impact Assessment (DPIA) has been developed, covering the FAIRVASC infrastructure and intended clinical research. In addition, a code of conduct has been developed for partners and project personnel as well as a standard operating procedure for onboarding and use. These living documents are available at:

Data Protection Impact Assessment (DPIA)

FAIRVASC Code of Conduct

FAIRVASC Standard Operating Procedure

6.1 Data Sharing Agreements, Codes of Practice and Standard Operating Procedures

To participate in FAIRVASC, any researchers' host institutions must sign up to a Data Sharing Agreement (DSA). The same is true for any Registries that are looking to be made available via the querying interface. The DSA is a Joint Controller Agreement (JCA) which ensures that each institution, whether they are a data provider or will be querying the data, or both, share in the control of FAIRVASC's data flows but also that their responsibilities remain clear in their activities with data.

The DSA/JCA provides links to the Codes of Practice and Standard Operating Procedures for use and onboarding as outlined above. To sign up to the DSA/JCA, please contact Nathan Lea at nathan.lea@i-hd.eu for further details.

6.2 Server Security and Low Cell Counts

The security of the triplestores must remain in the control of each Registry's host institution, where the base level of security is defined in the DSA/JCA, COP and SOP.

Whilst these specify the base security configurations, an additional protection mechanism has been placed into policy with regard to small cell counts. This means that any queries that return ten or fewer in terms of counts are obfuscated by saying that the results are 10 or fewer. This provides a balance between the richness of the results whilst limiting the risks of deductive disclosure.

More details on the rationale behind the security architecture and decisions are also available in the DPIA.

6.3 Sharing of the Data with Industry

Another data governance concern is the sharing of the registry data with third parties. As discussed above, FAIRVASC is now established as a platform that can be used to directly query multiple AAV registries in a streamlined fashion. This feature can make it appealing for third-party stakeholders, that is for entities that are not part of the FAIRVASC project but interested in the study of ANCA-associated vasculitis. A particular case of this is represented by for-profit private entities, especially the pharmaceutical industry. The data exchange between FAIRVASC and a pharmaceutical company might take two different forms: this might be a data sharing aimed at research or might be a trade where the data is sold to the company and thus put at their full disposal, with the company becoming the new data owner. In both scenarios, the exchange with the industry will be limited only to the data from those registries which provided explicit consent through signing a specific agreement. Onboarding registries as well need to state their stance toward the sharing of the data with industry. Thus, signing such an agreement is a prerequisite to onboard in the FAIRVASC project.

7 Data Quality

Quality data is frequently defined as data that is 'fit for purpose'. Data quality (DQ) is a developing field across the clinical, research and commercial worlds. In the research literature, some DQ concepts are poorly defined. Some DQ concepts also have multiple different names, which can lead to confusion. DQ is typically evaluated across specific measures known as 'domains', for example completeness, which refers to lack of missing data, and correctness, which refers to whether a value in a data set truly represents the real-life phenomenon it is intended to describe. Whilst it seems obvious that a good level of DQ should be a prerequisite for high quality biomedical research, there is minimal-to-no research that has been carried out to evaluate what levels of DQ are required for high quality research, or whether there are thresholds within DQ domains where the research output significantly improves when a particular level of DQ has been achieved. Regardless, DQ is increasingly recognised as important in the field of observational/epidemiological research. Therefore in FAIRVASC we have sought to establish a DQ process and culture of continuous DQ improvement that aims to maximise the quality of data used in the project.

Prior to data being uplifted into a local triplestore, it is required that the data undergo the FAIRVASC DQ analysis. The current form of the FAIRVASC DQ process is a straightforward analysis of four DQ domains. These domains are described in the table below. A data quality

analyst is required at each registry site. Statistical knowledge is not required, DQ analysts at registries have been clinicians or computer scientists. Basic counts and percentages are typically all that is required. Some DQ analysts have chosen to do analysis using a standard spreadsheet, while some have used their preferred statistical environment, such as R or python. A simple two-page work sheet has been used to guide the local DQ analyst on the required tasks. Results from the first round of DQ analysis were presented as a poster at the International ANCA and Vasculitis Workshop 2022, in Dublin. The poster can be seen at this link, on the fairvasc.eu website.

Uniqueness	The extent to which data lacks duplicates
Consistency	The extent to which data is consistent with expected formats (usually defined in the registry's data dictionary and ranges. For example, are dates in a standardised date format, ensure ages are numeric and do not contain alphabet characters, are biological values in a plausible range.
Completeness	Lack of missing data
Correctness	Does the data represent the real-world concept that it is intended to

Table 1 *FAIRVASC Data Quality domains*

8 Setting Up a Triplestore

8.1 Summary

The following is documentation of the processes involved for managing and implementing the R2RML mappings for the registry dataset. It also documents how that data is stored on a Fuseki triplestore running on a server, and how it can be accessed. These instructions are given assuming you are using a windows PC. For linux and MAC users, please adapt the R2RML processor commands appropriately.

8.2 Prerequisites and Resources

To execute the R2RML processor you will need the latest version of Java installed on your PC. For accessing the VMs, it is recommended you download putty and WinSCP (for windows users). The R2RML mappings can be accessed on GitHub (access must be granted). The R2RML processor is also available here. You can also find the harmonisation mappings in the Code Overview sheet.

8.3 The R2RML Mapping Files

The R2RML files are split into several separate files (but they can be provided in the same file as well). There are three main reasons for splitting the mappings:

1. The mapping files have a lot of content and so it is easier to manage them in a modular fashion. This also reduces the processing time when converting a large amount of data (e.g. when mapping the data for the disease activity score).

2. The mapping files reflect the three main types of mapping handled by FAIRVASC; Patient Overview, Patient Outcomes and Patient Encounters. This modular approach again makes it easier to manage the mappings, and the resulting RDF data generated.
3. Some of the mappings require pre-processing using SQL encoded in the R2RML mapping (such as filtering by diagnosis). For example; there are occasions where diagnosis is not present in rows which have data relevant to the mapping, and so, at times it has been necessary to separate the mappings in a way that filters are run in the appropriate triple map to reduce the generation of unwanted data.

8.4 Running the R2RML Processor to Generate RDF

To process the R2RML mappings with the R2RML processor and generate RDF from your tabular data, you must first edit the configuration file. The configuration file is called "config.properties" and has the similar content given in example below:

```
CSVFiles = RKD.csv
mappingFile = fairvasc_rkd_mapping_overview_v1.ttl
outputFile = fairvasc_rkd_output_overview_v1.ttl
format = TURTLE
```

The input files are the R2RML file (e.g. "fairvasc_rkd_mapping_overview_v1.ttl", that is your mappings to the harmonised schema) and the input CSV file (RKD.csv, your registry source data exported to a .csv) and the output file is "fairvasc_rkd_output_overview_v1.ttl" (e.g. the RDF data to be generated). Once the file is configured, you may use the run.bat file in the folder with the r2ml.jar file, the mapping file, the configuration file and the CSV file to generate the RDF. You can also run the contents of the batch file directly from the cmd line, e.g. java -Xmx12000m -jar r2rml.jar config.properties

```
java -Xmx12000m -jar r2rml.jar config.properties
```

The generated RDF will be located in the same folder as the config.properties file (unless you specify a different path than above).

8.5 Virtual Machine Server and Triplestore Hosting

You can configure a triplestore for the FAIRVASC data on a server. The storage space that is needed for the server must be gauged according to the size of the triplestore being uploaded, while as for the minimal RAM memory, 2 GB should roughly be a quite confident estimate. You can stop and start the triplestore as follows:

8.5.1 Starting/Stopping the Triplestores

To kill/stop and then restart each triplestore, you can use the following commands:

```
ps aux|grep fuseki  ##return process number of all fuseki instances
sudo kill -9 <process number> ##kill process
nohup java -Xmx1200M -jar fuseki-server.jar --port=3030 (any port you declare)
bg  ##to run the fuseki process in the background
```

nohup java -Xmx1200M -jar fuseki-server.jar --port=xxxx must be run from the appropriate folder, e.g. “apache-jena-fuseki-log-secure-proxy”, where the fuseki-server.jar file is located.

8.5.2 Configuring Authentication on Fuseki

Authentication in Fuseki is handled using the shiro.ini file. This can be found, in the case of the proxy triplestore, in apache-jena-fuseki-log-secure-proxy/run. The contents of the shiro.ini file are below. The main parts of interest for configuring usernames and passwords are highlighted. You can generate a password using any online password generator, e.g. https://passwordsgenerator.net/ We recommend using at least a 16 character password. Replace <password> with the generated password. This combined with username will be required to access the triplestore through the web client, and also within any queries to the endpoint.

```
# Licensed under the terms of http://www.apache.org/licenses/LICENSE-2.0

[main]
# Development
ssl.enabled = false

plainMatcher=org.apache.shiro.authc.credential.SimpleCredentialsMatcher
#iniRealm=org.apache.shiro.realm.text.IniRealm
iniRealm.credentialsMatcher = $plainMatcher

localhostFilter=org.apache.jena.fuseki.authz.LocalhostFilter

[users]
# Implicitly adds "iniRealm =  org.apache.shiro.realm.text.IniRealm"
admin=<password>
tcd_rkd=<password>
tcd_rkd=<password>

[roles]

[urls]
## Control functions open to anyone
/$/status  = anon
/$/ping    = anon
/$/metrics = anon
```

```
## and the rest are restricted to localhost.
# /$/** = localhostFilter
/$/** = authcBasic,user[admin]

# Endpoints restrictions
/** = authcBasic,user[admin]
```

8.5.3 Main Triplestore Details

The main triplestore which hosts the RDF data is hosted on the declared port (e.g. 3032). The best way to upload data is through the Fuseki web interface. Go to your web browser and on the server, type in the URL (e.g. http://www.example.com:3032/). This will take you to the web client. For starting and stopping, please see above instructions. Upload the generated RDF data from running the R2RML processor directly into the triplestore.

8.5.4 Configuring Logging on Fuseki

To enable logging, the file log4j2.properties must be added to the main folder , e.g. to the folder apache-jena-fuseki-log-secure-proxy for the proxy server. The contents of this file can be seen below. The highlighted part replaces the previous appender which writes logs to the console, and instead writes the logs into a file into the folder logs/log.fuseki.

```
## Licensed under the terms of http://www.apache.org/licenses/LICENSE-2.0
status = error
name = PropertiesConfig
filters = threshold

filter.threshold.type = ThresholdFilter
filter.threshold.level = ALL

##appender.console.type = Console
##appender.console.name = OUT
##appender.console.target = SYSTEM_ERR
##appender.console.layout.type = PatternLayout
#appender.console.layout.pattern = %d{HH:mm:ss:sss} %-5p %-15c{1} :: %m%n
##appender.console.layout.pattern = [%d{yyyy-MM-dd HH:mm:ss}] %-5p %-15c{1} ::
%m%n

## To a file.
appender.file.type = File
appender.file.name = FileLogger
appender.file.fileName=logs/log.fuseki
appender.file.layout.type=PatternLayout
appender.file.layout.pattern = [%d{yyyy-MM-dd HH:mm:ss}] %-5p %-15c{1} :: %m%n

rootLogger.level = debug
```

```
rootLogger.appenderRefs = file
rootLogger.appenderRef.file.ref = FileLogger

logger.jena.name  = org.apache.jena
logger.jena.level = INFO

logger.arq-exec.name  = org.apache.jena.arq.exec
logger.arq-exec.level = INFO

logger.arq-info.name  = org.apache.jena.arq.exec
logger.arq-info.level = INFO

logger.riot.name  = org.apache.jena.riot
logger.riot.level = INFO

logger.fuseki.name  = org.apache.jena.fuseki
logger.fuseki.level = INFO

logger.fuseki-fuseki.name  = org.apache.jena.fuseki.Fuseki
logger.fuseki-fuseki.level = INFO

logger.fuseki-server.name  = org.apache.jena.fuseki.Server
logger.fuseki-server.level = INFO

logger.fuseki-config.name  = org.apache.jena.fuseki.Config
logger.fuseki-config.level = INFO

logger.fuseki-admin.name  = org.apache.jena.fuseki.Admin
logger.fuseki-admin.level = INFO

logger.jetty.name  = org.eclipse.jetty
logger.jetty.level = WARN

# May be useful to turn up to DEBUG if debugging HTTP communication issues
logger.apache-http.name   = org.apache.http
logger.apache-http.level  = WARN

logger.shiro.name = org.apache.shiro
logger.shiro.level = WARN
# Hide bug in Shiro 1.5.0
logger.shiro-realm.name = org.apache.shiro.realm.text.IniRealm
logger.shiro-realm.level = ERROR

# This goes out in NCSA format
appender.plain.type = Console
appender.plain.name = PLAIN
appender.plain.layout.type = PatternLayout
```

```
appender.plain.layout.pattern = %m%n

logger.fuseki-request.name                 = org.apache.jena.fuseki.Request
logger.fuseki-request.additivity           = false
logger.fuseki-request.level                = OFF
logger.fuseki-request.appenderRef.plain.ref  = PLAIN
```

It is important to track this file while the server is running. Currently, copies are being made periodically and these are stored locally on the vm, but also can be found here -
These logs can be used to identify any unwanted queries being sent to the proxy triplestore. More information on how to set up a triplestore, run the R2RML processor and run queries in SPARQL can be found in the training material folders.

9 Contacts

For administrative queries of the onboarding to the FAIRVASC consortium please contact Michelangelo Tesi (mikitesi@gmail.com) for further details.

For queries regarding the data variables reused in the FAIRVASC consortium or questions about the disease captured in the FAIRVASC project please contact Karl Gisslander at karl.gisslander@med.lu.se.

For queries about data governance in the FAIRVASC project please contact Nathan Lea at nathan.lea@i-hd.eu for further details.

For queries regarding the data quality and the data quality steps undertaken by FAIRVASC registries please contact Michelangelo Tesi at mikitesi@gmail.com.

For queries regarding the technical implementation in FAIRVASC, the FAIRVASC interface, the FAIRVASC ontology or guidance in setting up a local triple store please contact Michelangelo Tesi at mikitesi@gmail.com.

10 Funding Acknowledgement

The FAIRVASC Project has received funding from the European Union's Horizon 2020 research and innovation programme under the EJP RD COFUND-EJP N° 825575.