HORIZON 2020
The EU Framework Programme for Research and Innovation

CAPABLE

Cancer Patients Better Life Experience

Grant Agreement No. 875052
Start Date: 01/01/2020 (48 Months)

Deliverable No. 3.2

Data-related functionality to realize a FAIR infrastructure

Due Date: [31/03/2021]
Submitted On: [16/06/2021]

Coordinator	University of Pavia (UNIPV)
Deliverable Lead Partner	AMC
Contributing Partners	UNIPV, UoH, BIOM, PUT, ICSM, NKI, DEON
Contact	Prof. Silvana Quaglini
Email	silvana.quaglini@unipv.it
Website	www.capable-project.eu

Deliverable Type		
R	Document, report	**X**
DEM	Demonstrator, pilot, prototype	
DEC	Websites, patent fillings, videos etc.	
OTHER		
Dissemination Level		
PU	Public	**X**
CO	Confidential (Consortium members including the Commission Services)	
CI	Classified Information (Commission Decision 2015/444/EC)	

This deliverable is a part of a project receiving funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 875052.

Table of Contents

List of Figures

List of Tables

1. Versions History

Version	Date	Author	Comments
0.5	March 16, 2021	R. Cornet	For internal review
0.7	April 25, 2021	R. Cornet	Reworking based on feedback from internal review
0.8	June 1, 2021	R. Cornet	Further elaboration on feedback
0.9	June 14, 2021	R. Cornet	Pre-final version in final layout.
1.0	June 16, 2021	R. Cornet	Final feedback processed.

2. Executive Summary

This deliverable analyzes and describes the **requirements** for establishing an infrastructure for CAPABLE that adheres to the Findability, Accessibility, Interoperability, Reusability (FAIR) Principles, with a focus on the data-related functionality. Hence, it describes the required functionality to adhere to the FAIR Principles. This analysis results in recommended implementation decisions for FAIR-enabling functionality.

This functionality is to be provided through the Application Programming Interfaces (APIs) of the CAPABLE components, and enables access to metadata, access to data, and data access control.

2.1. Requirements

To make CAPABLE data **Findable** a metadata repository for CAPABLE needs to be established, and this repository needs to be indexed in a searchable resource. To maximally adhere to the **Interoperability** principles, especially the possibility of linking to other (meta)data, this CAPABLE metadata repository should adhere to the FAIR Data Point (FDP) Specification [1]. This specification requires handling of metadata using Resource Description Framework (RDF). To provide access functionality to this RDF metadata, realization of a SPARQL Protocol and RDF Query Language (SPARQL)-endpoint is preferred, as this provides a protocol and query language dedicated to RDF. Adherence to the FDP Specification provides indexing of the CAPABLE FDP in the index of FAIR Data Points [2].

For CAPABLE data and metadata to fulfill the **Accessibility** principle, the CAPABLE metadata needs to remain available after the ending of the project, even if the data may not be accessible anymore at that time. To facilitate this, periodic deposition of metadata in a repository is required. FigShare [3] and Zenodo [4] are well-known examples of such repositories, and over 2000 other repositories can be found at the Registry of Research Data Repositories (re3data) [5]. Functionality to access metadata will be provided by the repository in which it is deposited, and by registries that index these repositories, such as the above-cited re3data, DataCite Commons [6], or OpenAIRE [7]. For metadata, no access conditions should apply.

While availability of CAPABLE data may end after the project, during the project data availability and accessibility will be essential to support data use within CAPABLE. This requires data access control (authentication and authorization) and an API to access the data. To ensure this API adheres to the **Reusability** principle of using domain-relevant community standards, HL7 FHIR [8] is recommended, being broadly adopted in healthcare, providing an elaborate model for representing the data, and enabling access control.

Given that metadata will be accessible without restrictions and will need to persist after the duration of the project, as opposed to data, for which access depends on authentication and authorization and may be time-limited, separation of the metadata repository from the data repository helps distinguishing these two types of data, while links between the metadata repository and the data repository provides integrated use. Further, for the metadata, use of domain-independent standards is preferred, as FAIR data stretches beyond the domain of healthcare. Conversely, for the CAPABLE data a domain-relevant standard is preferred, as this facilitates linking between CAPABLE data and third-party data.

2.2. Current status and next steps

HL7 FHIR is a healthcare data exchange standard that has reached a level of maturity that enables immediate uptake. This means that development of the data repository can be performed, where decisions will need to be made regarding the way in which the data are represented, i.e., modeled, as FHIR resources. Ontologies to be used in this modeling are available, and a core set of commonly used ontologies exists, so that the semantics of the exchanged data are explicit. While HL7 FHIR provides the functionality to exchange a broad range of healthcare data, including workflow and financial data, it doesn't provide a storage standard. This is overcome by adopting one of the mature data storage standards that

provide a mapping to HL7 FHIR. Among these is OMOP CDM, a generic data model with appropriate vocabularies, aimed at storing observational healthcare data. As CAPABLE focuses on such observational data, use of the OMOP CDM provides an adequately constrained model to represent the information being stored. Further, the OMOP model and vocabularies contribute to specification of a FHIR implementation profile, i.e., a set of relevant FHIR resources, which are constrained to uniformly and unambiguously describe the data to be exchanged. In other words, OMOP CDM provides a model that is adequate for the data being stored in CAPABLE, and that informs how to profile (i.e., constrain) FHIR resources for exchange of these data, where the semantics of the stored or exchanged data are sufficiently aligned and explicit.

Regarding metadata though, many questions are still open, as little consensus exists on what "rich metadata" should consist of, or at what level of granularity information should be provided. Further, no agreement exists on ontologies to use for representing this information, if such ontologies do indeed exist.

Consequently, currently, data exchange based on HL7 FHIR can be established, and this is under development in CAPABLE. For exchange of metadata, the community has not yet established agreement. Therefore, we suggest resorting to a generic FAIR Data Point, via which metadata are provided through SPARQL-queries, as these rely on RDF, and not on any specific models. This provides a flexible approach to provision of metadata, so that standards emerging during the CAPABLE project can be easily adopted and implemented.

3. Introduction

CAPABLE focuses on developing a cancer patient coaching system with the objective of facing the needs of people who have been treated for cancer. While this can be established by a closed, purpose-specific system, CAPABLE is committed to delivering a data management and software infrastructure that adheres to the FAIR (Findability, Accessibility, Interoperability, Reusability) principles, in order to increase the benefit of the data, information, and knowledge captured throughout the project.

Adoption of standard terminologies, data models, and APIs will enhance the Accessibility and Interoperability of these data, information, and knowledge so that it can be captured, exchanged, analyzed, and used during and after the CAPABLE project.

Addressing the FAIR principles right from the start and throughout the project rather than upon completion, influences design and implementation decisions that are made. This will remove the burden of data conversion at a later stage, and of software redesign.

This deliverable specifies the **functionalities** that enable (read only) interaction with data and metadata by humans and machines, i.e., the communication with the components within CAPABLE to query data and metadata.

In this section we outline what FAIR services pertain to, after which we address two complementary approaches to realize such services, SPARQL, in Section 4, and REST architecture in Section 5. Then, we assess the extent to which these approaches are FAIR in Section 6, based on which we specify our recommendation for the CAPABLE infrastructure in Section 7.

3.1. FAIR services

The realization of FAIR data is gaining steadily increasing attention since the specification of the FAIR data Principles in [9]. These FAIR principles are shown in the left column of Table 1.

The FAIR Principles are focused on specifying requirements to enable functionality to (first) Find and (if allowed) Access these data. This brings the challenge of assuming as little as possible a priori "knowledge" for humans and machines to find data and functionality. It is suggested that all information needed for humans and machines should be provided by so-called FAIR Digital Objects (FDO), i.e., "digital objects that fulfill all FAIR principles" [10]. These FDO "bind all critical information about an entity in one place and create a new kind of actionable, meaningful and technology-independent object" [11]. Currently, FDO provide a conceptual framework, but work is ongoing to also establish an implementation framework [12].

As can be seen in Table 1, Findability, Interoperability and Reusability are determined by the way in which data are represented, and by the metadata that is provided, and not by specific functionality. Only the Accessibility Guiding Principle A1 refers to the use of a "standardized communications protocol". This means that establishing FAIR services implies providing functionality in a way that adheres to such an open, free, universally implementable protocol, that takes into account authentication and authorization. A challenge is that, as goes for the other Principles, there are many ways in which adherence can be established.

FAIR Principles	**Required functionality to fulfil the principle**
To be Findable:	Searchable (third-party) repository, e.g.: ➢ FairDataPoint ➢ FigShare ➢ Zenodo (See section 7)
F1. (meta)data are assigned a globally unique and persistent identifier	No specific functionality
F2. data are described with rich metadata (defined by R1 below)	No specific functionality
F3. metadata clearly and explicitly include the identifier of the data it describes	No specific functionality
F4. (meta)data are registered or indexed in a searchable resource	Search-functions provided by (third-party) repository
To be Accessible:	Service with standardized communication protocol, such as HL7 FHIR, (based on REST) or SPARQL. (See sections 4 and 5)
A1. (meta)data are retrievable by their identifier using a standardized communications protocol	Query functionality
A1.1 the protocol is open, free, and universally implementable	Adopt standard protocol
A1.2 the protocol allows for an authentication and authorization procedure, where necessary	Protocol provides A&A
A2. metadata are accessible, even when the data are no longer available	No specific functionality
To be Interoperable:	Interoperability of service is established through criteria specified under Accessibility (standard protocol) and Reusability (community standards)
I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.	No specific functionality
I2. (meta)data use vocabularies that follow FAIR principles	No specific functionality
I3. (meta)data include qualified references to other (meta)data	No specific functionality
To be Reusable:	Provide guidance on usage and provenance
R1. (meta)data are richly described with a plurality of accurate and relevant attributes	No specific functionality
R1.1. (meta)data are released with a clear and accessible data usage license	To be selected in Deliverable 3.3

R1.2. (meta)data are associated with detailed provenance	Provenance functions based on data model and vocabularies (Section 6.1)
R1.3. (meta)data meet domain-relevant community standards	Adherence to data models and vocabularies (See sections 4 and 5.4)

Table 1. The FAIR Guiding Principles, accompanied by requirements regarding functionality of a FAIR infrastructure.

In this Deliverable we will introduce two complementary approaches to FAIR services that contribute to adherence to the A1 Principle for FAIR (meta)data and to FAIR services that are Interoperable. Then, we will assess how each approach contributes to adherence to the FAIR Principles, based on which recommendations are made.

1. Section 4 introduces the SPARQL communication protocol, which is rooted in the semantic web community, an important contributor to the FAIR ecosystem, especially contributing to findability (F1, F2, F3)
2. Section 5 introduces the Representational state transfer (REST) architecture, which is a broadly adopted approach for delivering web services. REST-services can be self-descriptive by using the OpenAPI specification [13], which is introduced in section 5.3, and contributes to accessibility (A1, A1.1)
   One of the main REST-based protocols used in healthcare is HL7 FHIR (Fast Healthcare Interoperability Resources), addressed in detail in section 5.4. Being RESTful, in HL7 FHIR each resource type has the same set of interactions defined that can be used to manage the resources in a highly granular way, which contributes to the interoperability of HL7 FHIR *services*, and to some extent to the interoperability of the data exchanged in HL7 FHIR (i.e., I1 and I2).

These two approaches are selected as they provide complementary input on how to deliver FAIR Data, and contribute to closing the gap between the FAIR Principles and implementation of actual services. In our assessment, we will focus on read-only functionality.

The next two sections first cover SPARQL, and then REST, including OpenAPI and HL7 FHIR.

3.2. FAIR Data Provenance

Provenance, introduced under the Reusability Principle in Table 1, requires functionality that is implemented using approaches such as SPARQL or REST, and dedicated models and vocabularies. The CAPABLE infrastructure, as healthcare at large, depends on reuse of data that has been captured previously, possibly at another site, in another information system, for another purpose, and using other representation standards. GO-FAIR provided the following explanation for "detailed provenance" [14]: "For others to reuse your data, they should know where the data came from (i.e., clear story of origin/history, see R1), who to cite and/or how you wish to be acknowledged. Include a description of the workflow that led to your data: Who generated or collected it? How has it been processed? Has it been published before? Does it contain data from someone else that you may have transformed or completed? Ideally, this workflow is described in a machine-readable format."

This description indicates that, as goes for many aspects of metadata, a realistic level of detail needs to be attained. Currently, there is no agreement on this level of detail, and we need to rely on high-level standards.

A short overview of Data Provenance Standards and Recommendations for FAIR Data can be found in [15]. At a high level, provenance can be based on the W3C PROV Data Model (PROV-DM) [16], depicted in Figure 1, which describes the relationships between activities associated with entities and the agents involved.

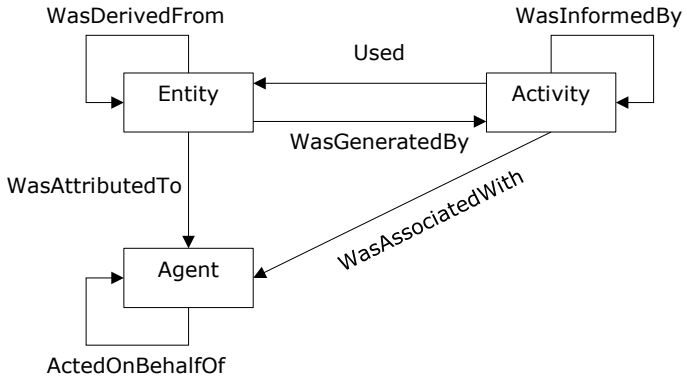

Figure 1. Core Structures of the PROV Data Model.

4. SPARQL

4.1. Introduction

SPARQL is a W3C (World Wide Web Consortium) recommendation, the current version being SPARQL 1.1 [17]. SPARQL stands for SPARQL Protocol and RDF Query Language. This name shows that it addresses two aspects. First, it provides a *specification* of a query language for RDF graphs [18]. Second, it specifies the *protocol* for implementation of this query language [19]. Part of the protocol is the SPARQL 1.1 Service Description [20]. In this way, SPARQL supports adherence to the Accessibility Principle A1, specifically A1.1.

SPARQL is designed for the Resource Description Framework (RDF). Being based on resources, RDF provides a major contribution to FAIR data by using Unique Resource Identifiers (URIs), as required by Findability Principle F1 "globally unique and persistent identifier". Persistence can be established by applying the 10 lessons described in [21], see also section 6.2.1.

Unique identifiers provide a means for fulfilling Interoperability Principles I1, I2, and I3, and by that the specification of rich metadata to represent explicit semantics by linking across RDF graphs, including both data and knowledge resources. As such, it enables "semantic interoperability". As an example with focus on Principle I2 "use vocabularies that follow FAIR principles", we specify that John Doe has a diagnosis of malignant melanoma. This is represented in RDF, using Turtle syntax, as follows:

```
https://example.com/johndoe
        https://semanticscience.org/resource/SIO_000217 [
                a http://snomed.info/id/439401001 ;
                https://semanticscience.org/resource/SIO_000008 [
                        a http://snomed.info/id/372244006
                ]
        ] .
```

Apart from the identifier for John Doe, being an example, all identifiers can be "resolved", i.e., information can be retrieved by following the links. From these links we can construct a pseudo-representation that is slightly more human-readable, still with the above URIs used as hyperlinks:

```
https://example.com/johndoe
        sio:hasQuality [
                a snomed:diagnosis ;
                sio:has_attribute  [
                        a snomed:malignant_melanoma
                ]
        ] .
```

Every URI renders these resources linkable. First, at the data level; as URIs are globally unique, all references to an identifier refer to the same resource, for example, John Doe. In this way information about John Doe can be merged, by gathering resources with this identifier from various RDF graphs. At the knowledge level, the semantics of "malignant melanoma" can be obtained from SNOMED CT, including its position in the (poly)hierarchy of SNOMED CT, and the characteristics that are specified.

4.2. SPARQL Query Language

The SPARQL Query Language resembles the Structured Query Language (SQL), both using SELECT … FROM … WHERE clauses. However, while the SQL syntax requires the use of the

"FROM" part, in which tables in the database are referenced, the "FROM" part in SPARQL is optional, referencing graphs that are accessible via the SPARQL endpoint. This means that unlike in SQL, where the table structure needs to be known, in SPARQL no such information is needed. This basic SPARQL-query demonstrates this principle:

```
select ?s ?p ?o where {?s ?p ?o}
```

This query specifies 3 variables to retrieve from an RDF graph those elements that satisfy the condition that a subject ?s has a predicate ?p with an object ?o. As any RDF graph is based on such subject-predicate-object triples, this query can be run on any SPARQL endpoint, and similar queries be used to inspect the contents of an RDF graph. This creates a level of transparency that is unmet by SQL, in which the tables and their attributes need to be known, as well as the relationships among those tables, i.e., the primary and foreign keys that are specified for this. While SQL implementations now commonly provide introspection of this structure through the information_schema, this adds a level of indirection that is circumvented in SPARQL. In this way, SPARQL adheres to Accessibility Principle A1, specifically A1.1 "the protocol is open, free, and universally implementable", where the protocol complements the query language and is described below.

4.3. SPARQL Protocol

The SPARQL Protocol "describes a means for conveying SPARQL queries and updates to a SPARQL processing service and returning the results via HTTP to the entity that requested them" [19].

In summary, it specifies that a SPARQL endpoint can be queried using HTTP GET or POST methods. When using the GET method, the SPARQL-query is provided in the "query" parameter.

As an example, this SPARQL-query can be used to retrieve person-diagnosis pairs:

```
SELECT ?person ?diagnosis WHERE { ?person sio:hasQuality [a snomed:diagnosis ;
sio:has_attribute  ?diagnosis ]}
```

It would be executed with this GET-request:

```
GET /sparql/?query= SELECT%20%3Fperson%20%3Fdiagno-
sis%20WHERE%20%7B%20%3Fperson%20sio%3AhasQuality%20%5Ba%20snomed%3Adiagno-
sis%20%3B%20sio%3Ahas_attribute%20%20%3Fdiagnosis%20%5D%7D HTTP/1.1
Host: www.example.com
User-agent: my-sparql-client/0.1
```

The response provided is in one of these formats: SPARQL XML Results Format, SPARQL JSON Results Format, SPARQL CSV/TSV Results Format, or an RDF serialization.

The result of the example query above would be:

```
HTTP/1.1 200 OK
Date: Fri, 06 May 2005 20:55:12 GMT
Server: Apache/1.3.29 (Unix) PHP/4.3.4 DAV/1.0.3
Connection: close
Content-Type: application/sparql-results+xml

<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">

 <head>
   <variable name="person"/>
   <variable name="diagnosis"/>
 </head>
 <results>
   <result>
     <binding name="person"><uri>http://www.example/per-
son/johndoe</uri></binding>
     <binding name="diagnosis"><bnode>r29392923r2922</bnode></binding>
   </result>
...
</sparql>
```

This shows that the XML-response adheres to the "sparql-results" schema, in which each result consists of bindings of URIs or so-called blank nodes to the variables specified in the query.

The SPARQL Protocol does not specify mechanisms for authentication and authorization, and hence relies on HTTP(S) mechanisms for that.

4.4. SPARQL Service Description

According to [20], SPARQL services made available via the SPARQL Protocol should return a service description document at the service endpoint when dereferenced using the HTTP GET operation without any query parameter strings provided. Such a service description provides additional information about the capabilities or configuration of a SPARQL endpoint. Such capabilities may relate to the protocol, and can for example specify the result format:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:sd="http://www.w3.org/ns/sparql-service-description#">
  <sd:Service>
    <sd:endpoint rdf:resource="http://www.example/sparql/"/>
    <sd:supportedLanguage rdf:resource="http://www.w3.org/ns/sparql-
service-description#SPARQL11Query"/>
    <sd:resultFormat rdf:resource="http://www.w3.org/ns/for-
mats/RDF_XML"/>
    <sd:resultFormat rdf:resource="http://www.w3.org/ns/formats/Tur-
tle"/>
  </sd:Service>
</rdf:RDF>
```

The response above shows that this endpoint supports SPARQL 1.1 Query language, and can return results in RDF/XML or Turtle.

4.5. SPARQL Limitations

Choosing a "semantic web" approach, using RDF and SPARQL, will contribute to adherence to a significant part of the FAIR Principles, given the unique identification and interlinking of resources contribute to rich, explicit metadata, and the openness of SPARQL. However, there are a number of caveats that require attention.

While the FAIR Principles focus on the data, they pay little attention to the processes of data collection and use. For data collection, there is generally a need to constrain the attributes that are relevant for certain entity types, and what the domain of each attribute is. This is for example what is specified in a database by the table structures, or in an electronic Case Report Forms (eCRF). The specification of such a so-called schema is required to render the data that are made accessible via SPARQL also interoperable and reusable. This means that domain-relevant community standards are adopted and represented using RDF-based schema languages, such as ShEx (Shape Expressions) [22] or ShaCL (Shapes Constraint Language) [23]. In the domain of healthcare, such representations are not yet commonly available.

Another limitation is that SPARQL does not yet provide adequate integration with some of the other semantic web languages, such as the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL). OWL provides the representational basis for powerful inferencing capabilities over ontologies, while SWRL enables inferencing by means of rules. Both languages enrich sets of axioms with additional, inferred, axioms.

For example, with this SWRL rule a "hasUncle" attribute does not need to be stated, but will be inferred for those individuals who have a parent who has a given brother:

```
hasParent(?x1,?x2) ∧ hasBrother(?x2,?x3) ⇒ hasUncle(?x1,?x3)
```

However, querying over or retrieval of such inferred properties in SPARQL cannot be performed in real-time. It would require preprocessing of a dataset, i.e., determining inferred properties, and adding these to the dataset.

4.6. SPARQL - Summary

The above analysis shows that SPARQL is a highly generic query language and protocol, which provides some room for refinement. This means that there is little possibility or need for configuration, so that general-purpose clients can be used to communicate with a SPARQL endpoint. A further advantage of SPARQL is that it is designed for RDF, rendering the data to be accessed, being based on URIs, in principle at the highest possible level of interoperability.

While ShEx or ShaCL schemas are still scarce in the domain of healthcare, they do exist for metadata. Examples thereof are the metadata schema DCAT (Data Catalog Vocabulary) and its Application Profile for data portals in Europe (DCAT-AP) [24]. The latter is implemented in the EU Open Data Portal, which provides a SPARQL-endpoint [25].

In Section 6 this will be further assessed and related to the RESTful approaches that is addressed in the next section.

5. REST architecture

5.1. Introduction

REST, which stands for Representational State Transfer, is a HTTP-based architecture for client-server interaction [26]. While the range of such interactions is broad, in this Deliverable we will restrict our assessment to the scope of data-related interaction, mainly querying data in a RESTful way.

The basis of REST is the use of URLs to get a representation of the state of a target resource. To retrieve (i.e., get a representation of the state of) a patient with id 1, a RESTful request would be:

```
GET /Patient/1 HTTP/1.1
Host: www.example.com
User-agent: my-rest-client/0.1
```

The resource that is returned is commonly represented in XML or JSON (Javacript Object Notation). A JSON-example would be:

```
{
    "resourceType": "Patient",
    "identifier": [
        {
            "system": "http://hl7.org/fhir/sid/us-ssn",
            "value": "1"
        }
    ],
    "fullURL": "http://www.example.com/Patient/1",
    "name": [
        {
            "use": "official",
            "family": "Doe",
            "given": [
                "John",
                "J.D."
            ],
            "prefix": [
                "Mr."
            ]
        }
    ],
    "gender": "male",
    "birthDate": "1955-05-05"
}
```

This small example shows that there are both commonalities and differences in the requests and responses in SPARQL and REST, on which we will reflect later.

5.2. RESTful services – architectural constraints

Within the scope of data-related functionality, a number of characteristics of RESTful services are of relevance, which relate to the "Uniform Interface" constraint [27].

5.2.1. Resource identification in requests

As can be seen in the example in the introduction, the requested resource is identified in the request, in the example above /Patient/1. Similar to SPARQL, in this way resources are made linkable, by providing their reference as a URI. This is elaborated on in section 5.2.3.

5.2.2. Self-descriptive messages

RESTful messages must include the information needed to describe how the message is to be processed. This includes specification of, for example, the media type provided in a response.

5.2.3. Hypermedia as the engine of application state (HATEOAS)

As mentioned in section 5.2.1, RESTful servers need to provide links to enable clients to discover related resources. This is known as HATEOAS [28]. This implies that those resources must be identified by these links, and may be identified in other ways. In the example in 5.1, the "fullURL" provides this identification, in this case a self-reference, in addition to the "identifier" specified.

Further, a RESTful server must provide responses that include hyperlinks to other resources that are currently available, where the availability may depend on the state of a resource. For example, only if a patient is admitted there will be admission information to provide, such as a location or department where the patient is admitted.

5.3. RESTful API Description Language

While the REST architecture requires messages to be self-descriptive, there is no constraint stating that RESTful servers need to be self-descriptive. That is, unlike SPARQL, that includes a service description, for RESTful servers or APIs such descriptions may be missing. This means that prior knowledge is needed regarding the available resources. In the example in 5.1, one needs to know that a "Patient" resource is available.

However, various description languages exist, of which a list of 15 can be found in [29]. From these, the OpenAPI Specification [13] is emerging as a broadly adopted and actively maintained language, of which the latest release, version 3.1.0 dates February 2021. Like SPARQL service description, OpenAPI provides a declarative resource specification, so that clients can communicate with the server without prior knowledge.

The OpenAPI Specification addresses, among others, the following aspects:

1. Schemes: http and/or https.
2. Paths: all paths that are available to clients, including their methods (GET, POST, PUT, DELETE) and required or optional parameters, as well as the types of produced responses.
3. Security definitions: means to access resources, such as api keys or authorization.
4. Definitions: these provide the models, i.e., the schemas to which the resources adhere.

Generally, this specification is provided both via a graphical user interface, to enable interactive exploration of server capabilities, and as a JSON-file, to be consumed by client applications.

While such specification is intended to enable clients to communicate without prior knowledge, a challenge that remains is that the specification is fully at the syntactic level.

This is illustrated by the example below. This shows that the semantics of "patient" are not made explicit in the specification, hence interpretation of the label is required to determine the exact semantics.

```
"/patient/{patientId}": {
  "get": {
    "tags": [
      "patient"
    ],
    "summary": "Find patient by ID",
    "description": "Returns a single patient",
    "operationId": "getPatientById",
    "produces": [
      "application/json",
      "application/xml"
    ],
    "parameters": [
      {
        "name": "patientId",
        "in": "path",
        "description": "ID of patient to return",
        "required": true,
        "type": "integer",
        "format": "int64"
      }
    ],
    "responses": {
      "200": {
        "description": "successful operation",
        "schema": {
          "$ref": "#/definitions/Patient"
        }
      },
      "400": {
        "description": "Invalid ID supplied"
      },
      "404": {
        "description": "Patient not found"
      }
    }
  }
```

5.4. HL7 FHIR

As mentioned, RESTful services are highly generic and (should) expose their model via an API Description Language, which is largely syntactic. Hence, for truly interoperable services it is essential to agree on a common resource model for which the semantics of the elements in it are made maximally explicit.

5.4.1. Resources and profiles

In the healthcare domain, such a common model is provided by HL7 FHIR [8]. It provides a specification of those resources that represent the medical domain, such as Patient, Practitioner, Care Team, Allergy, Problem, Procedure, to mention a few of the approximately 150 resources, of which an overview can be found in [30].

A graphical representation of a fragment of the specification of the Patient resource is shown in Figure 2.

Together these resources constitute an elaborate model of information to be exchanged in healthcare. This model is however deliberately permissive, i.e., it poses as little as possible constraints on the resources, as can be seen for example in Figure 2 by the minimum cardinality of 0, meaning that practically each attribute can be disregarded in an implementation. In a practical communication use case it is useful to be stricter, and for example enforce the presence of certain attributes. This is facilitated by means of so-called profiles.

Apart from the specification, APIs or Software Development Kits (SDKs) for HL7 FHIR have been made available, which facilitate implementing specification-compliant services. Examples of such APIs are:

- HAPI FHIR [31], a free and open source Java API
- SMART on FHIR [32], which provides libraries for a variety of programming languages, including Python and Swift (IOS).
- FIRELY .NET SDK [33], the official support SDK for working with HL7 FHIR on the Microsoft .NET (dotnet) platform.

Name	Flags	Card.	Type	Description & Constraints
Patient	N		DomainResource	Information about an individual or animal receiving health care services Elements defined in Ancestors: id, meta, implicitRules, language, text, contained, extension, modifierExtension
identifier	Σ	0..*	Identifier	An identifier for this patient
active	?! Σ	0..1	boolean	Whether this patient's record is in active use
name	Σ	0..*	HumanName	A name associated with the patient
telecom	Σ	0..*	ContactPoint	A contact detail for the individual
gender	Σ	0..1	code	male | female | other | unknown AdministrativeGender (Required)
birthDate	Σ	0..1	date	The date of birth for the individual
deceased[x]	?! Σ	0..1		Indicates if the individual is deceased or not
deceasedBoolean			boolean	
deceasedDateTime			dateTime	

Figure 2. Fragment of the specification of the HL7 FHIR Patient resource.

5.4.2. Structure Definitions and Capability Statements

While the standard HL7 FHIR resources provide a common interpretation of their semantics and an agreed upon specification for use, profiles, although essential for practical implementation, introduce the risk of breaking interoperability. This risk is minimized by using resources for the specification of profiles, specifically the StructureDefinition resource. Availability of this resource enables client-side discovery of the profiles used, and hence establishes a means for determining the extent to which given profiles are compatible with profiles from other FHIR servers.

Apart from configuring individual resources by the use of profiles, FHIR servers may implement only a fragment of the full set of resources. For example, finance-related resources may be out of scope in a certain scenario. This means though, that clients need to be able to detect which resources are available, and which methods are allowed, i.e., whether they are read-only (i.e., http GET), or read/write (i.e., http POST or PUT). Available resources and methods are represented as a CapabilityStatement resource.

FHIR servers provide a capability statement via the /metadata URI. This enables clients to retrieve the exact set of resources, profiles, and allowed methods that are exposed via the server. In this way, a service description is provided at the level of the models, which is a refinement of the service description specified in the HL7 FHIR standard.

6. Assessment of approaches for a FAIR infrastructure

In the preceding sections two approaches were introduced that can contribute to providing functionality for a FAIR infrastructure for CAPABLE in different ways, first SPARQL in Section 4, then REST and its healthcare-specific standard HL7 FHIR in Section 5.

In this section we compare the functionality provided by SPARQL and HL7 FHIR, separating functionality for data from that for metadata where relevant.

6.1. Comparison of approaches

To come to a substantiated recommendation, we assess how both approaches relate to each of the FAIR Principles.

F1. (meta)data are assigned a globally unique and persistent identifier

- SPARQL is based on RDF and hence on globally unique resource identifiers for data and metadata.
- HL7 FHIR can adhere to this principle by applying consistent resource identification for data and metadata, as described in the HL7 FHIR specification [34].

➢ CAPABLE's FAIR infrastructure can use SPARQL and/or HL7 FHIR.

F2. data are described with rich metadata (defined by R1 below)

- SPARQL nor HL7 FHIR require or enforce description with rich metadata for findability, such as a description of data origin, authorship, inclusion criteria.
- No specific FHIR Resources exist to provide such a description.
- SPARQL can provide such descriptions in RDF by adopting schemas such as Dublin Core [35], DCAT, or DataCite Metadata Schema [36]. See section 6.2.1 for more details.

➢ CAPABLE's FAIR infrastructure can use SPARQL for metadata, applying proper schemas.

F3. metadata clearly and explicitly include the identifier of the data it describes

- As mentioned in F2, HL7 FHIR will in general not be able to cover required metadata.
- SPARQL can represent metadata, but explicit identification of described data will depend on implementation.

➢ CAPABLE's FAIR infrastructure can use SPARQL for metadata, relating to the identifiers of relevant data, which could be provided by HL7 FHIR or SPARQL.

F4. (meta)data are registered or indexed in a searchable resource

- Given the sensitive nature of the data, only metadata will be indexed.

➢ CAPABLE's FAIR infrastructure can use SPARQL to index metadata in a searchable resource. Such a searchable resource can be a project-specific or third-party FAIR Data Point, complemented by a repository such as FigShare [3] or Zenodo[4], which safeguards persistence of metadata after the end of the project. (Also see Principle A2)

A1. (meta)data are retrievable by their identifier using a standardized communications protocol
- SPARQL provides such a protocol for data and metadata.
- HL7 FHIR provides such a protocol for data.

➢ CAPABLE's FAIR infrastructure can use SPARQL for retrieval of metadata and data, and HL7 FHIR for retrieval of data.

A1.1 the protocol is open, free, and universally implementable
- Both SPARQL and HL7 FHIR comply with this Principle.

A1.2 the protocol allows for an authentication and authorization procedure, where necessary
- Access to metadata will not require authentication and authorization.
- Access to data will require authentication and authorization, which HL7 FHIR provides. SPARQL relies on HTTP(S) methods for authentication and authorization.

➢ CAPABLE's FAIR infrastructure can use HL7 FHIR for retrieval of data.

A2. metadata are accessible, even when the data are no longer available

➢ See Principle F4: CAPABLE's FAIR infrastructure can use SPARQL to index metadata in a searchable resource.

I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation
- SPARQL, with RDF underpinning it, meets this Principle. OWL is an even more formal knowledge representation language, but does not fully integrate with SPARQL (see section 4.5).
- HL7 FHIR itself is not a knowledge representation language, but can make use of vocabularies and ontologies that meet this Principle. Further, in addition to the commonly used XML and JSON formats, FHIR provides representation of resources as RDF triples, using the Multi-purpose Internet Mail Extensions (MIME)-type `application/fhir+turtle`.

➢ CAPABLE's FAIR infrastructure can use SPARQL to represent data and metadata in a formal way, and can use HL7 FHIR to represent data, using formal vocabularies and ontologies as much as possible.

I2. (meta)data use vocabularies that follow FAIR principles
- Both SPARQL and HL7 FHIR implementations can make use of such vocabularies, provided that the extent to which vocabularies follow FAIR principles is properly described or assessed. For SPARQL, subjects, predicates as well as objects use such vocabularies, whereas for HL7 FHIR these vocabularies are predominantly used to represent objects, i.e., the values specified.

➢ CAPABLE's FAIR infrastructure can be based on such vocabularies for both data and metadata, using SPARQL and/or HL7 FHIR.

I3. (meta)data include qualified references to other (meta)data
- In SPARQL, qualified references, i.e., attributes or links that are semantically rich (e.g., " has biological role" instead of "depends on") can be included by using dedicated vocabularies such as the Open Biological and Biomedical Ontologies (OBO) Relation Ontology [37].
- HL7 FHIR specifies many references for the resources. These references are well-qualified, but their definitions are provided in free text, and not based on a formal representation.

➢ CAPABLE's FAIR infrastructure can use SPARQL for delivering metadata and data using qualified references based on relevant ontologies. HL7 FHIR can be used for qualified references, which are not, but could be related to formal ontologies.

R1. meta(data) are richly described with a plurality of accurate and relevant attributes

- The descriptions of data and metadata are in itself metadata. As described for F2 and F3 above, HL7 FHIR will in general not be able to provide such descriptions.

➢ CAPABLE's FAIR infrastructure can use SPARQL to provide rich descriptions, applying proper schemas that provide the required accurate and relevant attributes.

R1.1. (meta)data are released with a clear and accessible data usage license

- Data use conditions can be expressed using for example the Data Use Ontology (DUO) [38].

➢ CAPABLE's FAIR infrastructure can use SPARQL to provide this information.

R1.2. (meta)data are associated with detailed provenance

- Provenance, as briefly described in section 3.2, addresses the description of the origin(s) of as well as the processing performed on data and metadata. To specify such metadata, the W3C PROV Data Model (PROV-DM) [16] can be used, especially with the related PROV Ontology (PROV-O) [39], which is an OWL2 ontology that allows for mapping of the PROV data model to RDF.
- To represent provenance of individual FHIR resources, the FHIR Provenance Resource [40] can be used.

➢ CAPABLE's FAIR infrastructure can use SPARQL and PROV-O to provide provenance information for metadata such as datasets, and use the HL7 FHIR Provenance Resource for provenance of data.

R1.3. (meta)data meet domain-relevant community standards

- In the domain of healthcare, the adoption of HL7 FHIR is expected to be widespread by 2024 [41]. So it is expected to be a contemporary but future-proof healthcare data exchange standard.
- For metadata, SPARQL can meet this Principle, provided that relevant metadata standards as introduced in this deliverable are used. As metadata standards is a broad area in which there are many developments, in part driven by the FAIR Principles, standards need to be selected carefully, especially as agreement needs to be reached [Repository Features to Help Researchers: An invitation to a dialogue [42].

➢ CAPABLE's FAIR infrastructure can use SPARQL for retrieval of metadata, and HL7 FHIR for retrieval of data.

We see that although some differences exist in the adherence to FAIR Principles of both approaches, it highly depends on the data and metadata being provided, and the way in which this is done. HL7 FHIR and other RESTful services lack the use of a formal language for knowledge representation (I1). It has a better authentication and authorization mechanism (A1.2), as this can be applied on a per-resource basis, whereas SPARQL relies on allowing or restricting access to the endpoint. FHIR, via its rich and broadly adopted model, has the potential of providing rich references to other data and metadata (I3).

This assessment makes clear that, although differences are limited, in the healthcare domain it makes sense to adhere to the RESTful architecture of HL7 FHIR, given its model being broadly adopted in the healthcare community. Further, it is important to realize that this does not imply an impossibility to benefit from the strength that SPARQL provides, mainly in its explicit semantics. Efforts are ongoing to enable RDF-representations of FHIR resources, and representations of their specifications are available. These schema representations are provided in the RDF-based language Shape Expression (ShEx) language, a schema language for RDF, and can be found at [43]. However, currently support for ShEx is still limited, and the strength of formal knowledge representation in SPARQL may be impacted by practical issues related to large-scale reasoning over sets of larger expressive ontologies, and especially over large datasets, as well as the intricacies of correct modelling, as demonstrated for example at [44].

Conversely, efforts are ongoing to enable a RESTful way to interact with SPARQL endpoints. One example hereof is grlc [45], a service for converting SPARQL queries (stored in GitHub repositories) to RESTful APIs, including OpenAPI specifications.

To conclude, while ultimately a SPARQL endpoint with enhanced security features might result in the most strictly FAIR implementation of the CAPABLE infrastructure, the currently overly large modeling freedom and implementation challenges render this a longer-term solution, as overcoming these would introduce more uncertainty than acceptable. Hence, implementing the **CAPABLE FAIR data infrastructure as a set of HL7 FHIR services**, with as much as possible linking to high-quality vocabularies, explicit *specification of metadata*, and keeping up to date with the semantic web aspirations within the FHIR community is an approach that is not only likely to be successful in the shorter term, but also FAIR enough in the long run.

For *specification of metadata*, **a SPARQL endpoint can implement the CAPABLE FAIR metadata infrastructure**, as this will provide domain-independent access to these metadata.

6.2. From Requirements to Implementation

The requirements for data-related functionality to realize a FAIR CAPABLE infrastructure were assessed taking into account the FAIR principles as well as standard approaches towards FAIR functionality. As this is an area in which progress and community adoption of standards may be different, and design decisions taken will introduce additional decisions to be made, we perform a brief, and inevitably incomplete, analysis of the impact of the proposed choices.

6.2.1. SPARQL for metadata-related functionality

Representation of metadata using RDF and made available via a SPARQL-endpoint requires decision making regarding the software application to be used and the models to adhere to.

Implementing SPARQL

Various generic applications exist that provide such a SPARQL-endpoint for RDF data. This includes:

- **Apache Jena Fuseki** [46], a free open source SPARQL server.
- **Blazegraph DB** [47], a free SPARQL server.
- **OpenLink Virtuoso** [48], a free open source SPARQL server.

The SPARQL-endpoint can be referenced in a FAIR Data Point (FDP), using for example the Python implementation [49], or the Java implementation [50]. In this way, the endpoint gets registered in the FDP home [2], through which the endpoint becomes Findable. To ensure persistence of metadata, deposit metadata in a repository such as Zenodo.

Metadata schemas for SPARQL

As mentioned, RDF and SPARQL don't impose any restrictions on the data, which is in this case metadata, that is represented. This means that a schema needs to be determined to which this metadata adheres. Various schemas are of relevance, which can be used in combination if needed:

- Dublin Core Metadata Initiative [35]. This schema provides a set of metadata terms, including properties, vocabulary encoding schemes, syntax encoding schemes, and classes.
- Data Catalog Vocabulary (**DCAT**) and its Application Profile for data portals in Europe (**DCAT-AP**) [24], which was briefly introduced in Section 4.6. DCAT provides a layered schema for representing among others the following types of resources, cf. [51]:
  - Catalogue: a dataset in which each individual item is a metadata record describing some resource; the scope of dcat:Catalog is collections of metadata about datasets or data services.

- Dataset: a collection of data, published or curated by a single agent. A dataset can be delivered as one or more distributions, and/or via a dataservice.
- Distribution: an accessible form of a dataset such as a downloadable file.
- Data Service: a collection of operations accessible through an interface (API) that provide access to one or more datasets or data processing functions.

- DataCite [36]. The DataCite Metadata Schema provides the core metadata properties for accurate and consistent identification of a resource, as well as recommended use instructions. DataCite is intended to be generic to the full range of research datasets, and should be complemented by discipline or community-specific metadata to fully describe the data, in order to make it reusable.
- Data Use Ontology [38]. This provides a framework to describe data use conditions. See section 6.1, under Principle R1.1.
- W3C PROV Data Model (PROV-DM) [16] and related PROV Ontology (PROV-O) [39]. See section 6.1, under Principle R1.2.

Specification of metadata in SPARQL – use of identifiers

The first FAIR principle states that "(meta)data are assigned a globally unique and persistent identifier". While this may seem straightforward, there are various challenges to do this properly. 10 "lessons" are provided in [21], as guidance for implementation, as mention in section 4.1.

1. Credit any derived content using its original identifier
2. Help local IDs travel well: Document prefix and patterns
3. Opt for simple, durable web resolution
4. Avoid embedding meaning or relying on it for uniqueness
5. Design new identifiers for diverse uses by others
6. Implement a version-management policy
7. Do not reassign or delete identifiers
8. Make URIs clear and findable
9. Document the identifiers you issue and use
10. Reference and display responsibly

An exhaustive inventory and analysis of applications, schemas and implementation recommendations is beyond the scope of this deliverable, but the above options provide an indication of feasibility of delivering the required functionality to establish CAPABLE's FAIR infrastructure that adheres as much as possible to existing standards.

6.2.2. HL7 FHIR for data-related functionality

HL7 FHIR services or endpoints are implemented in a large number of organizations, for demonstration, development and everyday use in clinical practice. As addressed in section 5.4.1, various APIs are available to provide FHIR-compliant services, including HAPI FHIR and the FIRELY .NET SDK. While these APIs provide the functionality addressed in this deliverable, they do not cater for actual storage of the data. Various options exist to implement this storage:

- **Fhirbase** [52], an open source toolkit for storing and working with FHIR data, using a PostgreSQL database.
- **OMOP CDM** [53], the Observational Medical Outcomes Partnership Common Data Model, a highly generic data model, linked to over 50 healthcare vocabularies. Mapping between OMOP CDM and HL7 FHIR is facilitated for example through the OMOP-on-FHIR toolkit [54]. Like Fhirbase, OMOP CDM is also generally implemented in a PostgreSQL database. The model and vocabularies contribute to specification of a FHIR implementation profile.

7. Recommendations

From the analysis and the assessment performed in the previous sections, recommendations are formulated to establish a FAIR infrastructure for CAPABLE.

At a high level, the main recommendations are:

- **Deliver CAPABLE metadata using RDF and SPARQL.** As metadata are largely domain-independent, this generic approach provides access to metadata for researchers from any domain, without introducing the need of understanding domain-specific APIs.
- **Provide access to CAPABLE data, including authentication and authorization, using HL7 FHIR.** This approach ensures that data are delivered in a way broadly accepted by the healthcare community, and prevents implementation of proprietary models. Using OMOP CDM as underlying storage model further ensures use of broadly accepted vocabularies.

These high-level recommendations can further be detailed as specified in Table 2, which provides references to the relevant FAIR principles for each of the recommendations, and also addresses data-related requirements.

Recommendation	Related FAIR principles
1. Create a CAPABLE FAIR Data Point (FDP), e.g., according to the implementation provided at [50] or the Python implementation at [49]. Using that implementation contributes to F4, as the FDP will be registered at the FDP home [2].	F4, A1, A1.1, A2, I1
2. For metadata, use globally unique persistent identifiers taking into account the 10 lessons described in [21].	F1
3. Populate the CAPABLE FDP with sufficiently rich metadata, including a description of the dataset based on the Data Catalog Vocabulary (DCAT) [51] or its Application Profile for data portals in Europe (DCAT-AP) [24].	F2, F3
4. Metadata in the CAPABLE FDP adopts established vocabularies used in the legal, research, and medical domain.	I2, I3, R1.3
5. Provide license and provenance information, using Data Use Ontology [38], Provenance Data Model [16], and Provenance Ontology [39].	R1, R1.1, R1.2
6. Provide access to data using HL7 FHIR.	A1, A1.1, A1.2
7. Model data provided by the HL7 FHIR server as much as possible using established vocabularies used in the medical domain. Representation of resources using Turtle syntax can be considered, especially if mappings between resource types (and their attributes) and other ontologies are established.	I2, I3, R1.3
8. The CAPABLE FDP provides reference to the HL7 FHIR server.	F3, R1

Table 2. Recommendations.

8. Glossary

API	Application Programming Interface
CDM	Common Data Model
DCAT	Data Catalog
DUO	Data Use Ontology
eCRF	electronic Case Report Forms
EOSC	European Open Science Cloud
FAIR	Findability, Accessibility, Interoperability, Reusability
FDP	FAIR Data Point
FHIR	Fast Healthcare Interoperability Resources
HATEOAS	Hypermedia as the engine of application state
HL7	Health Level 7
HTTP	HyperText Transfer Protocol
MIME	Multi-purpose Internet Mail Extensions
OBO	Open Biological and Biomedical Ontologies
OMOP	Observational Medical Outcomes Partnership
OWL	Web Ontology Language
PROV-DM	PROV Data Model
PROV-O	PROV Ontology
RDF	Resource Description Framework
re3data	Registry of Research Data Repositories
REST	Representational State Transfer
SDK	Software Development Kit
ShEx	Shape Expressions
SIO	Semanticscience Integrated Ontology
SPARQL	SPARQL Protocol and RDF Query Language
SWRL	Semantic Web Rule Language
W3C	World Wide Web Consortium

9. References

[1] FAIR Data Team, *FAIR Data Point design specification*. https://github.com/FAIR-DataTeam/FAIRDataPoint-Spec (accessed Jun. 14, 2021).
[2] *FAIR Data Points Index*. https://home.fairdatapoint.org/ (accessed Jun. 13, 2021).
[3] *FigShare*. https://figshare.com/ (accessed Jun. 13, 2021).
[4] European Organization For Nuclear Research and OpenAIRE, *Zenodo*. https://zenodo.org/ (accessed Jun. 13, 2021).
[5] *Registry of Research Data Repositories*. https://www.re3data.org/ (accessed Jun. 13, 2021).
[6] *DataCite Commons*. https://commons.datacite.org/
[7] *OpenAIRE*. https://www.openaire.eu/
[8] *HL7 FHIR*. https://www.hl7.org/fhir/
[9] M. D. Wilkinson *et al.*, "The FAIR Guiding Principles for scientific data management and stewardship," *Sci. Data*, vol. 3, no. 160018, 2016, doi: 10.1038/sdata.2016.18.
[10] U. Schwardmann, "Digital Objects – FAIR Digital Objects: Which Services Are Required?," *Data Sci. J.*, vol. 19, no. 1, p. 15, doi: 10.5334/dsj-2020-015.
[11] *FAIR Digital Objects Forum*. https://fairdo.org/ (accessed Jun. 13, 2021).
[12] *FDO Technical Specification & Implementation Group*. https://fairdo.org/wg/fdo-tsig/ (accessed Jun. 13, 2021).
[13] SmartBear Software, *OpenAPI Specification*. https://swagger.io/specification/ (accessed Jun. 13, 2021).
[14] *FAIR Principle R1.2: (Meta)data are associated with detailed provenance*. https://www.go-fair.org/fair-principles/r1-2-metadata-associated-detailed-provenance/ (accessed Jun. 13, 2021).
[15] M.-L. Jauer and T. M. Deserno, "Data Provenance Standards and Recommendations for FAIR Data." doi: 10.3233/shti200380.
[16] L. Moreau and P. Missier, *PROV-DM: The PROV Data Model*. https://www.w3.org/TR/prov-dm/ (accessed Jun. 13, 2021).
[17] The W3C SPARQL Working Group, *SPARQL 1.1 Overview*. https://www.w3.org/TR/sparql11-overview/ (accessed Jun. 13, 2021).
[18] S. Harris and A. Seaborne, *SPARQL 1.1 Query Language*. https://www.w3.org/TR/2013/REC-sparql11-query-20130321/ (accessed Jun. 13, 2021).
[19] L. Feigenbaum, G. T. Williams, K. G. Clark, and E. Torres, *SPARQL 1.1 Protocol*. https://www.w3.org/TR/sparql11-protocol/ (accessed Jun. 13, 2021).
[20] G. T. Williams, *SPARQL 1.1 Service Description*. https://www.w3.org/TR/2013/REC-sparql11-service-description-20130321/ (accessed Jun. 13, 2021).
[21] J. A. McMurry *et al.*, "Identifiers for the 21st century: How to design, provision, and reuse persistent identifiers to maximize utility and impact of life science data," *PLOS Biol.*, vol. 15, no. 6, p. e2001414, Jun. 2017, doi: 10.1371/journal.pbio.2001414.
[22] E. Prud'hommeaux, I. Boneva, J. E. Labra Gayo, and G. Kellogg, *Shape Expressions Language 2.1*. http://shex.io/shex-semantics/ (accessed Jun. 13, 2021).
[23] H. Knublauch and D. Kontokostas, *Shapes Constraint Language*. https://www.w3.org/TR/shacl/ (accessed Jun. 13, 2021).
[24] European Commission, *DCAT Application Profile for data portals in Europe*. https://joinup.ec.europa.eu/collection/semantic-interoperability-community-semic/solution/dcat-application-profile-data-portals-europe/release/201-0 (accessed Jun. 13, 2021).
[25] *SPARQL Endpoint of the official portal for European data*. https://data.europa.eu/sparql/ (accessed Jun. 13, 2021).
[26] R. T. Fielding, *Architectural styles and the design of network-based software architectures*. Irvine: University of California, 2000.
[27] *Representational state transfer*. https://en.wikipedia.org/wiki/Representational_state_transfer (accessed Jun. 13, 2021).
[28] *HATEOAS*. https://en.wikipedia.org/wiki/HATEOAS (accessed Jun. 13, 2021).

[29] *Overview of RESTful API Description Languages*. https://en.wikipedia.org/wiki/Overview_of_RESTful_API_Description_Languages (accessed Jun. 13, 2021).
[30] *HL7 FHIR Resource Index*. https://www.hl7.org/fhir/resourcelist.html (accessed Jun. 13, 2021).
[31] *HAPI FHIR*. https://hapifhir.io/ (accessed Jun. 13, 2021).
[32] *SMART on FHIR*. https://docs.smarthealthit.org/ (accessed Jun. 13, 2021).
[33] *Firely .NET SDK for HL7 FHIR*. https://github.com/FirelyTeam/firely-net-sdk (accessed Jun. 13, 2021).
[34] *HL7 FHIR Base Resource Definitions - Consistent Resource Identification*. https://www.hl7.org/fhir/resource.html#consistency (accessed Jun. 13, 2021).
[35] DCMI Usage Board, *DCMI Metadata Terms*. https://www.dublincore.org/specifications/dublin-core/dcmi-terms/ (accessed Jun. 13, 2021).
[36] *DataCite Metadata Schema*. https://schema.datacite.org/ (accessed Jun. 13, 2021).
[37] OBO, *Relation Ontology*. http://www.obofoundry.org/ontology/ro.html (accessed Jun. 13, 2021).
[38] *Data Use Ontology*. http://www.obofoundry.org/ontology/duo.html (accessed Jun. 13, 2021).
[39] T. Lebo, S. Sahoo, and D. McGuinness, *PROV-O: The PROV Ontology*. https://www.w3.org/TR/prov-o/ (accessed Jun. 13, 2021).
[40] *HL7 FHIR Resource Provenance*. https://www.hl7.org/fhir/provenance.html (accessed Jun. 13, 2021).
[41] C. Jason, *API Adoption Slow, Widespread FHIR Uptake Expected by 2024*. https://ehrintelligence.com/news/api-adoption-slow-widespread-fhir-uptake-expected-by-2024 (accessed Jun. 13, 2021).
[42] M. Cannon *et al.*, *Repository Features to Help Researchers: An invitation to a dialogue*. https://doi.org/10.5281/zenodo.4084762 (accessed Jun. 13, 2021).
[43] *HL7 FHIR - Patient.shex*. https://www.hl7.org/fhir/patient.shex.html (accessed Jun. 13, 2021).
[44] *SNOMED CT - for instance(s)*. https://www.youtube.com/watch?v=iktqgCzt77Y (accessed Jun. 13, 2021).
[45] *grlc*. https://grlc.io/ (accessed Jun. 13, 2021).
[46] *Apache Jena*. https://jena.apache.org/ (accessed Jun. 13, 2021).
[47] *Blazegraph Database*. https://blazegraph.com/ (accessed Jun. 13, 2021).
[48] *Virtuoso Open-Source Edition*. http://vos.openlinksw.com/owiki/wiki/VOS (accessed Jun. 13, 2021).
[49] *Python implementation of FAIR Data Point*. https://github.com/fair-data/fairdata-point (accessed Jun. 13, 2021).
[50] *FAIR Data Point*. https://github.com/FAIRDataTeam/FAIRDataPoint (accessed Jun. 13, 2021).
[51] R. Albertoni, D. Browning, S. Cox, A. Gonzalez Beltran, A. Perego, and P. Winstanley, *Data Catalog Vocabulary (DCAT) - Version 2*. https://www.w3.org/TR/vocab-dcat-2/ (accessed Jun. 13, 2021).
[52] *Fhirbase - Open Source Database for HL7 FHIR*. https://www.health-samurai.io/fhirbase (accessed Jun. 13, 2021).
[53] *OMOP Common Data Model*. https://www.ohdsi.org/data-standardization/the-common-data-model/ (accessed Jun. 13, 2021).
[54] *OMOP on FHIR - Implementation of mapping between OMOP and FHIR*. https://github.com/omoponfhir (accessed Jun. 13, 2021).